PROPERTY OF
HEWLETT
PACKARD
FINANCE AND REMARKETING DIVISION

solartron
instruments

SI3535D, F & G
Scorpio
Data Logger

REFERENCE MANUAL

Digitized by the Internet Archive
in 2025

https://archive.org/details/manualsplus_07256

solartron
instruments

SI3535D, F & G
Scorpio Data Logger

REFERENCE MANUAL

Solartron Instruments is a division of the Solartron Group Ltd. Any reference to Schlumberger or Schlumberger Instruments (the name of the company until November 1993) implies no liability on the part of Schlumberger.

Solartron Instruments
a division of the Solartron Group Ltd
Victoria Road, Farnborough,
Hampshire GU14 7PW. UK.

Issue CA : January 1994
Solartron Part No. : 35352007

19©94

Solartron Instruments
Victoria Road, Farnborough
Hampshire GU14 7PW England.
Telephone 0252 376666.
Fax: 0252 544981

Solartron Instruments
11261 Richmond Avenue, Suite #G-105
Houston, Texas, 77082-2615 USA
Telephone: (713) 558 4930
Fax: (713) 558 4701

Solartron Instruments
37 Rue du Saule Trapu
91882 Massy, Cedex, France
Telephone: (1) 69 53 63 53
Fax: (1) 60 13 37 06

Solartron Overseas Sales Ltd
Instruments Division
Block 5012 TECHplace II
Ang Mo Kio Ave. 5, #04-11
Ang Mo Kio Industrial Park
Singapore 2056,
Republic of Singapore
Telephone: (65) 4823500
Fax: (65) 4824645

Solartron Instruments
Room 7006, Yuan Shan Hotel
No. 20 Yu Min Tong Li
West District, Beijing 100029
China
Telephone: (86) 1-2029133
Fax: (86) 1-2028617

For details of our agents in other countries, please contact our Farnborough, UK, office.

Solartron Instruments pursue a policy of continuous development and product improvement.
The specification in this document may therefore be changed without notice.

GENERAL SAFETY PRECAUTIONS

The equipment described in this manual has been designed in accordance with IEC publication 348 (Class 1) Safety Requirements for Electronic Measuring Apparatus, and has been supplied in a safe condition. To avoid injury to an operator or service technician the safety precautions given below, and throughout the manual, must be strictly adhered to, whenever the equipment is operated, serviced or repaired. For specific safety details, please refer to the relevant sections within the manual.

The equipment is designed solely for electronic measurement and should be used for no other purpose. Solartron Instruments accept no responsibility for accidents or damage resulting from any failure to comply with these precautions.

GROUNDING
To minimize the hazard of electrical shock it is essential that the equipment is connected to a protective ground whenever the power supply, measurement or control circuits are connected, even if the equipment is switched off. The protective ground for ac and dc supplies is connected separately.

AC GROUND is connected via the ac supply cord. The cord must be plugged into an ac line outlet with a protective ground contact. When an extension lead is used, this must also contain a ground conductor. Always connect the ac supply cord to the supply outlet before connecting the control and signal cables; and, conversely, always disconnect control and signal cables before disconnecting the ac supply cord. The ac ground connection must have a continuous current rating of 6A.

DC GROUND is connected via a ground stud on the equipment power supply unit (PSU). The dc ground connection must have a continuous current rating of 35A.

Where both protective grounds are used it must be ensured that these grounds are, and will remain, at the same potential.

AC SUPPLY VOLTAGE
Never operate the equipment from a line voltage or frequency in excess of that specified. Otherwise, the insulation of internal components may break down and cause excessive leakage currents.

FUSES
Before switching on the equipment check that the fuses accessible from the exterior of the equipment are of the correct rating. The rating of the ac line fuse must be in accordance with the voltage of the ac supply.

Should any fuse continually blow, do not insert a fuse of a higher rating. Switch the equipment off, clearly label it "unserviceable" and inform a service technician.

EXPLOSIVE ATMOSPHERES
NEVER OPERATE the equipment, or any sensors connected to the equipment, in a potentially explosive atmosphere. It is NOT intrinsically safe and could possibly cause an explosion.

Continued overleaf.

SAFETY PRECAUTIONS (continued from previous page)

SAFETY SYMBOLS

For the guidance and protection of the user, the following safety symbols appear on the equipment:

SYMBOL	MEANING
	Refer to operating manual for detailed instructions of use.
	Hazardous voltages.
	Protective conductor terminal. This must be connected to ground before operating the equipment.

NOTES, CAUTIONS AND WARNINGS

For the guidance and protection of the user, Notes, Cautions and Warnings appear throughout the manual. The significance of these is as follows:

NOTES	highlight important information for the reader's special attention.
CAUTIONS	guide the reader in avoiding damage to the equipment.
WARNINGS	guide the reader in avoiding a hazard that could cause injury or death.

AVOID UNSAFE EQUIPMENT

The equipment may be unsafe if any of the following statements apply:

- Equipment shows visible damage.
- Equipment has failed to perform an intended operation.
- Equipment has been subjected to prolonged storage under unfavorable conditions.
- Equipment has been subjected to severe physical stress.

If in any doubt as to the serviceability of the equipment, don't use it. Get it properly checked out by a qualified service technician.

LIVE CONDUCTORS

When the equipment is connected to its measurement inputs or supply, the opening of covers or removal of parts could expose live conductors. The equipment must be disconnected from all power and signal sources before it is opened for any adjustment, replacement, maintenance or repair. Adjustments, maintenance or repair, must be done only by qualified personnel, who should refer to the Maintenance Manual.

EQUIPMENT MODIFICATION

To avoid introducing safety hazards, never install non-standard parts in the equipment, or make any unauthorized modification. To maintain safety, always return the equipment to Solartron Instruments for service and repair.

SI3535 Reference Manual Contents

Chapter

1 GENERAL INFORMATION

1 General Information 1.3
2 SI3535 System Variants 1.3
3 SI3535 Command Language 1.5
4 SI3535 Processing Program Language 1.5
5 How to use this manual 1.6

2 LANGUAGE CONFIGURATION

1 Language Configuration 2.3

3 SI3535 COMMAND LANGUAGE

1 Introduction 3.3
2.1 Initialization Command 3.5
2.2 Acknowledge Command 3.8
2.3 Channel Definition Commands 3.9
2.3.1 Analog Input Channel 3.11
2.3.2 Analog Output Channel 3.14
2.3.3 Decimal-type Digital Input Channel 3.16
2.3.4 Status-type Digital Input Channel 3.19
2.3.5 Event-type Digital Input Channel 3.21
2.3.6 Counter-Timer Input Channel 3.23
2.3.7 Channel Definition Example 3.25
2.4 Conversion Definition Command 3.26
2.5 Text Definition Command 3.27
2.6 Task Definition Command 3.28
2.6.1 Measurement Task 3.29
2.6.2 Event Task 3.35
2.6.3 Task Definition Example 3.40
2.7 Renumber Command 3.41
2.8 Drift Command 3.41
2.9 Time Tag Command 3.41
2.10 Digital IO Card Reset Enable/Disable Command 3.42
2.11 Bargraph Command 3.42
2.12 Template Command 3.43
2.13 Trends Command 3.44
2.14 Disk Command 3.48
2.15 Monitor Command 3.52
2.16 Write Message Command 3.53
2.17 Measure Command 3.54
2.18 List Command 3.55
2.19 Set Command 3.56
2.20 Control Commands 3.57
2.21 Query Commands 3.59
2.22 Transient Capture Command 3.62
2.23 Export Command 3.70

Chapter

4 SI3535 PROCESSING LANGUAGE

1	Introduction	4.3
2	Processing Programs	4.4
3	Program Constants and Variables	4.10
4	Control Structures	4.17
5	Output Statements	4.20
6	Mathematical Operators and Functions	4.31
7	Entering and Editing Programs	4.34
8	Digital Inputs	4.39
9	Task Cross.Control	4.41
10	Counters and Timers	4.42
11	Random Channel Access	4.44
12	Storing and Replaying Processed Data and Programs	4.46
13	Memory Capacity	4.47
14	Processing Programming Techniques	4.48
15	Example Programs	4.49

5 OUTPUT FORMAT

1	General Information	5.3
2	Output Data Formats	5.5
3	Coded Binary Formatting	5.15
4	Binary Format	5.25
5	Data Blocking	5.33
6	Status Variables	5.34
7	Storing Data on Disk (SI3535 D & G)	5.36

6 SERIAL INTERFACE

1	General Information	6.3
2	Standard RS232 Interface	6.4
3	Fast Serial Interface	6.16
4	Front Panel Emulation	6.24

7 GPIB INTERFACE

1	General Information	7.3
2	GPIB Status Designations	7.3
3	GPIB Protocol	7.4
4	GPIB Address	7.5
5	Service Requests	7.6
6	Polling	7.8
7	Input/Output Data	7.8
8	Communication via GPIB Bus	7.8
9	GPIB Interface Connections	7.9

Appendices

1 Command Summary.
2 Sensor Number Tables.
3 Error Messages.

Chapter 1
Introduction

Section		*Page*
1	**General Information**	1.3
2	**SI3535 System variants**	1.3
2.1	Interface Options	1.3
3	**SI3535 Command Language**	1.5
4	**SI3535 Processing Program Language**	1.5
4.1	Processing Facilities Summary	1.5
5	**How to use this manual**	1.6

Illustrations

1 1	General peripheral interconnections	1.4

1 GENERAL INFORMATION

The SI3535 comprises a versatile and flexible data acquisition and logging facility, with powerful software processing capabilities. Details are given in this manual of the Command Language for definition and control, and Processing Programming Language for post-processing. Refer to the User guide for information on the operation of the Scorpio Data Logger.

Set-up routines and measured data can be stored by means of a disk drive incorporated within the SI3535D and SI3535G, and a printer provides hardcopy of program listings as required.

2 SI3535 SYSTEM VARIANTS

The SI3535 is available in the following three variants, each capable of being controlled by the Command Language and programmed for Task Processing SCORPIO can be set up and controlled either locally, from the front panel, or remotely using a computer or other intelligent peripherals connected to one of the remote interfaces. An RS232 interface is supplied as standard, RS422 and GPIB interface modules are available as options.

Three versions of the SCORPIO data logger are available:

SI3535D: has a front panel keyboard and display. Setting up may be performed using the keyboard in conjunction with the display which offers menus from which the operator may make the required selection. A printer and a disk drive are built in.

SI3535F & G: these two units have a 'blank' front panel, and are intended for remote control. A VT220 or local VDU connected to the monitor interface socket on the front panel or a suitable device connected to one of the rear interfaces, may be used.

The SCORPIO SI3535G includes a floppy disk drive, allowing storage on a remotely controlled system.

The general terms SI3535 and SCORPIO are used throughout this Manual when describing functions available to all three versions of the data logger. However, when describing functions that are specific to a single version of logger, the terms SI3535D, SI3535F or SI3535G have been used as appropriate.

2.1 INTERFACE OPTIONS

The SCORPIO can be controlled locally, using the Front panel keys of SI3535D, or via a terminal attached to the local interface socket of an SI3535F and G, using the command language.

The SCORPIO may also be controlled remotely, using a computer, or other intelligent peripherals, connected to the standard or one of the optional interface sockets located on the SI3535 back panel.

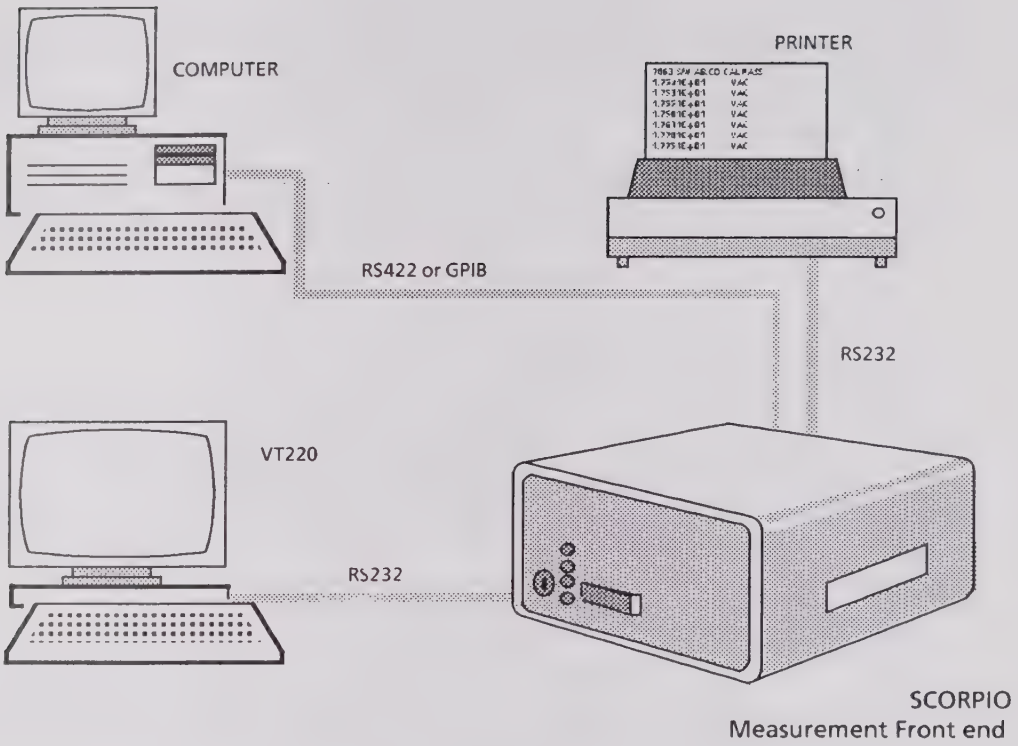


Fig 1.1 *General peripheral interconnections*

The following interface options are available with the SI3535:

RS232 Interface

An RS232 socket on the controller board is supplied as standard and is suitable for connection to a computer, or any device capable of controlling the SCORPIO via an RS232 link. In addition the socket may be used to connect an output only recording device, such as a printer.

Fast Serial, RS422/232 (BS4505)

The Fast serial interface board, 35304B, is an optional board having two output ports that may be set independently for either RS422 or RS232 operation, with BS4505 protocol on one of them. In addition limited RS232 protocol is available.

GPIB

The GPIB interface board, 35304A, fully implements the GPIB interface standard and will support any GPIB compatible device such as a computer or output peripheral. SCORPIO may be set up to be a Talker/listener device or Talker only device.

3 SI3535 COMMAND LANGUAGE

The SI3535 Command Language is based on English words, and has been designed for both flexibility and ease of use. For each command word, two characters are recognized by SI3535 as a command mnemonic.

The language facilitates the entering of channel information and task definition set-up. Up to eight tasks can be set up and run either individually, or concurrently as required.

4 SI3535 PROCESSING PROGRAM LANGUAGE

The Processing Language is similar to BASIC, and enables programs to be written for manipulation of results obtained during scanning. Independent processing programs can be set up for all eight scanning tasks, with global variables allowing values to be passed between tasks.

The Processing facility is commonly used to create derived results from real channel measurements. A complete range of arithmetic and trigonometric operators are available as standard, allowing complex calculations to be performed Some examples of processing programs used are task cross-triggering, data reduction, and output control by means of analog or digital signals.

4.1 PROCESSING FACILITIES SUMMARY

The facilities available using a SI3535 processing program are summarized as follows:

a) Modification of individual channel results by various mathematical operations such as scaling and conversions (these operations can augment or replace the conversion facilities available under Channel Definition).

b) Calculations involving cross-channel results, time averages and control algorithms.

c) Adding explanatory text to recorded results.

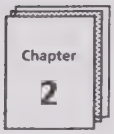
d) Directing selected results to the printer, display and peripheral interfaces.

e) Cross-triggering and cross-aborting of tasks.

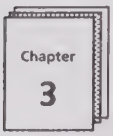
f) Decision processing utilizing combinations of logical conditions, applied to analog and digital results.

5 HOW TO USE THIS MANUAL

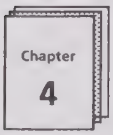
The purpose of this Manual is to provide reference information for the Command and Processing Languages, which can be used with the SI3535 and the interface facilities available. The scope and content of the topics within each chapter is outlined as follows:



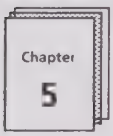
Gives details of the Language configurations and variants available.



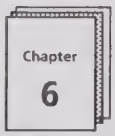
Provides the command rules, syntax and tree structures for the SI3535 Command Language.



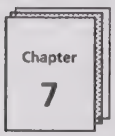
Provides reference information for the SI3535 Processing Language and details the facilities available.



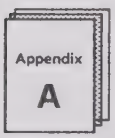
Provides information about the output formats and gives details concerning the disk drive / disk compatibility.



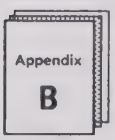
Gives information concerning the RS232 serial interface (controller), RS232/422 fast serial (35304B module) and the RS232 front panel emulation.



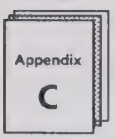
Gives information concerning the GPIB interface module.



Appendix A shows all command language tree structures for quick reference.



Appendix B contains tables with the sensor number codes detailed.



Appendix C provides a list of all error codes and messages.

Index

Chapter 2
Language Configuration

Section *Page*

1 Language Configuration 2.3
1.1 Language Variations 2.3

Tables

2.1 Language Configuration Settings 2.3

1 LANGUAGE CONFIGURATION

The SI3535 can be configured to operated in one of several Language modes. This principally effects the front panel Menu system, but also effects the presentation of dates and numbers.

The Default Language is determined by the setting of two switches on the Controller module. The switches are read by the SI3535 on power-up, and during a device clear. The current language can be configured or queried using the SI3535 command language as follows:

a) The Command language syntax to query the current language setting is:

IN LA ?

The Reply to this command is

BR - for British, AM - for American, FR - for French
GE - for German

b) The remote command to configure the language is:

IN LA <LG>, where <LG> is substituted by one of the above language mnemonics.

The Default Configuration, as determined by the rear panel switch on the Controller module, is set according to Table 2.1.

Table 2.1 *Language Configuration Settings*

Language	Controller Module	
	Switch 4	Switch 3
British English	On	On
American English	On	Off
French	Off	On
German	Off	Off

1.1 LANGUAGE VARIATIONS

There are a few differences between the language variants for the SI3535. These are listed below :

Front Panel Menus and Softkeys : use appropriate language

Date Formats : In American English mode, the date format is mm-dd-yy for both input and output. All other languages use dd-mm-yy.

Number Format : The decimal point marker is a "." (fullstop) for British English and American English, and a "," (comma) for French and German.

Where two non-integers must be entered (as in conversion command), the separator between them should be a ";" (semi-colon). However, for British English and American English, the "," (comma) may also be used. This maintains compatibility with Orion.

Temperatures : For American English, temperatures are output in °F. For all other languages, °C will be used. However conversions are available to alter the units if you prefer.

Chapter 3
SI3535 Command Language

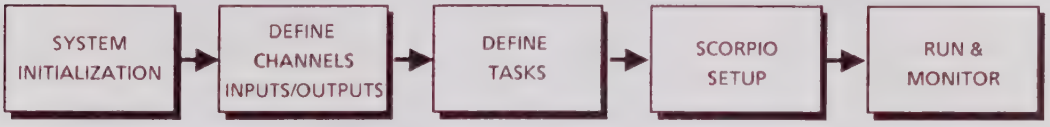
Section		*Page*
1	**Introduction**	3.3
1.1	Command Tree Structures	3.3
1.2	Command Syntax	3.3
1.3	Sending Commands to the SI3535	3.4
1.4	Entering Commands	3.4
2.1	Initialization Command	3.5
2.2	Acknowledge Command	3.8
2.3	Channel Definition Commands	3.9
2.3.1	Analog Input Channel	3.11
2.3.2	Analog Output Channel	3.14
2.3.3	Decimal.type Digital Input Channel	3.16
2.3.4	Status.type Digital Input Channel	3.19
2.3.5	Event.type Digital Input Channel	3.21
2.3.6	Counter-Timer Input Channel	3.23
2.3.7	Channel Definition Example	3.25
2.4	Conversion Definition Command	3.26
2.5	Text Definition Command	3.27
2.6	Task Definition Command	3.28
2.6.1	Measurement Task	3.29
2.6.2	Event Task	3.35
2.6.3	Task Definition Example	3.40
2.7	Renumber Command	3.41
2.8	Drift Command	3.41
2.9	Time Tag Command	3.41
2.10	Digital IO Card Reset Enable/Disable Command	3.42
2.11	Bargraph Command	3.42
2.12	Template Command	3.43
2.13	Trends Command	3.44
2.14	Disk Command	3.48
2.15	Monitor Command	3.52
2.16	Write Message Command	3.53
2.17	Measure Command	3.54
2.18	List Command	3.55
2.19	Set Command	3.56
2.20	Control Commands	3.57
2.21	Query Commands	3.59
2.22	Transient Capture Commands	3.62
2.23	Export Commands	3.70

1 INTRODUCTION

The SI3535 Command Language can be used either for front panel control of a SI3535D, local terminal control of SI3535F and G, or remote control of all three versions. When controlling the SI3535 from a remote interface, it is possible to set-up, interrogate or modify the database.

There is also the capability of running, halting and aborting tasks, or monitoring measurements on specific SI3535 channels.

The SI3535 Command Language can be sub-divided into the following areas:



Initialization	Channel Definition	Task Definition	Drift correct	Monitor
Acknowledge	Conversion Definition	Processing	Time Tag	Measure
	Text Definition		Disk	List
			Barcharts (3535D)	Run/Halt..etc
			Templates (3535D)	Query

In addition to the features described above, when defining a task to run on the SI3535, the processing facility can be selected This facility allows users to extend the processing capability of the task being defined and is described in Chapter 4.

1.1 COMMAND TREE STRUCTURES

Related SI3535 commands are grouped together, to give a tree-like structure referred to as a Command Tree. The command trees define all the possible options associated with a specific function (eg. definition of an event task, single channel measurements). Many of the commands shown in a specific tree are optional, and the users only need to enter the options required.

1.2 COMMAND SYNTAX

Each command consists of a two letter mnemonic followed by a qualifying parameter, (a numeric or alpha/ numeric variable).

eg. **TA 3 OP ME**

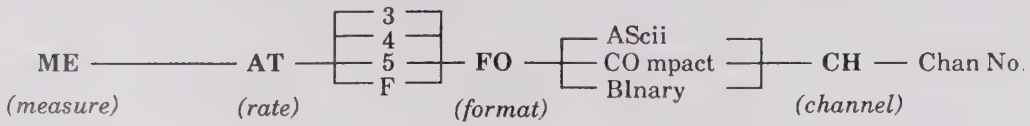
TA : Command Mnemonic [for TAsk]
3 : Numeric variable
OP : Command mnemonic [for OPeration]
ME : Option variable [for MEasure]

1.3 SENDING COMMANDS TO THE SI3535

When sending commands, refer to chapters 6 and 7 for information on how to send commands to the various interfaces. A command or group of commands is normally sent to the SI3535 as a command string. A command mnemonic is used to define the tree structure of which subsequent commands in the string form a part; these comprise any valid commands. The string is finally terminated with C/R, when all related commands have been entered.

A typical example is given for sending a command to the SI3535 which takes a single measurement on a specified channel:

The command tree syntax for the MEASURE COMMAND is:



ME : Command mnemonic that defines the Measure Command Tree. ME must be the first command in the input string, and requires no qualifying parameter.

{ **AT**, **FO**, **CH** } : Any of these commands from the tree can be entered next.

To send a command string to instruct the SI3535 to take a single measurement on channel 10 (previously defined) and output the result in full ASCII format, enter command:

ME FO AS **CH**10 C/R

Note : Command **AT** is omitted, and default value F, or the value used for any previous measure command is used.

Error messages are provided by the SI3535 for errors in the command string. Refer to Appendix 3 for details.

1.4 ENTERING COMMANDS

The following should be observed when sending command strings to the SI3535:

a) Command string must start with the mnemonic defining the command tree, and can only contain one tree-defining command.

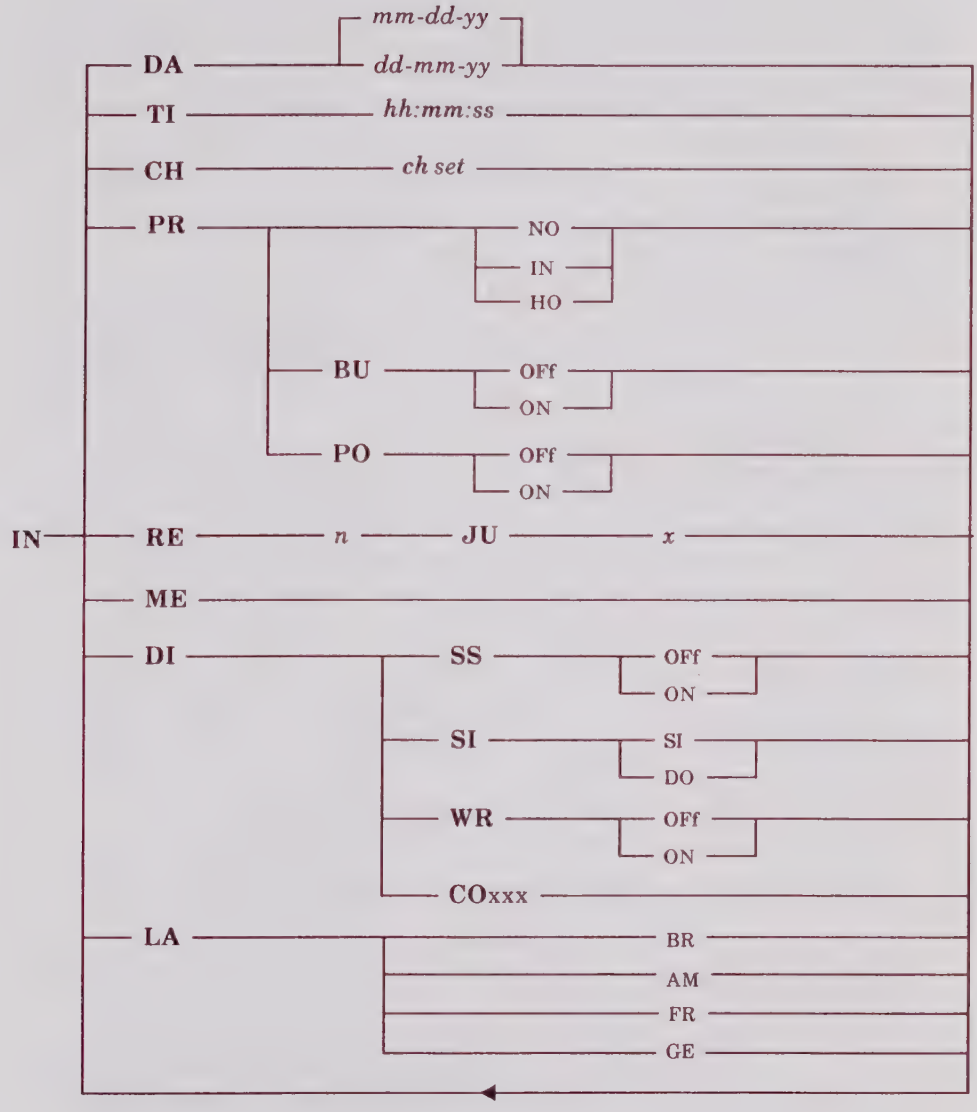
b) A delimiting space must be entered between a command and an alpha variable or option. A space is optional if the variable is numeric.

c) The total number of characters in a command string must not exceed 80 characters, including delimiting spaces.

d) If a command and its parameter is entered more than once, the last value received is used.

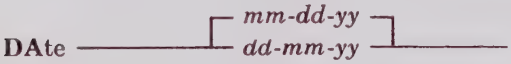
2.1 INITIALIZATION COMMAND

COMMAND TREE STRUCTURE:



Command mnemonics for the Initialization Command Tree are as follows:

Function **IN** : Initialize date, time, printer etc



Function **DA** : Set up current date in internal clock

dd-mm-yy current date
mm-dd-yy current date (American English)

TIme ——— *hh:mm:ss* ———

Function **TI** : Set up current time held in internal clock.

hh:mm:ss Time

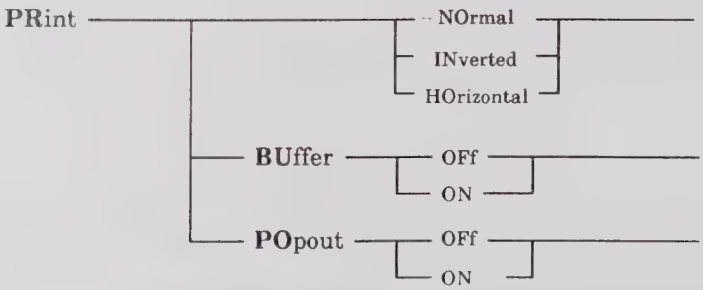
Note 1 : Seconds are optional
Note 2 : A SI3535 may be synchronized to an external computer using the SY? query command.

CHannel ——— *ch set* ———

Function **CH** : Initialize analog channels specified: this measures and stores 'unstrained' parameters from strain gauge channels and sets up auto ranging.

ch set channel numbers to be initialized.

Note 1 : CH x IN ? where x is a single channel. This causes the instrument to return a floating point number. This can be entered to initialize strain gauges to a specific value using the command CH x IN <FP NUM >



Function **PR** : Initialize printer (SI3535D only) to print in normal or inverted mode.

NO	Normal - top reading is most recent data.
IN	Inverted - bottom reading is most recent data.
HO	Horizontal - prints out screen dumps horizontally.

BU Printer Buffer. When ON, if the printer is unable to print, due to the print head being lifted, or no paper left, any output to the printer is buffered and the printer icon on the front panel flashes. When the printer is able to resume the buffered output is printed. If buffering is OFf, data sent to the printer, when it is unable to print, will be discarded.

PO Printer Popout. When printer popout is on, if there is no output to the printer for about 3 secs, the paper is automatically advanced to enable the last item printed to be viewed. When the next item is to be printed, the paper is rewound to its original position. Note that this is known as View Mode in front panel menus.

REference ——— *n* ——— **JUnction** —— *x*

Function **RE** : Select external thermocouple reference junctions.

n 1 to 4: number of reference junction.

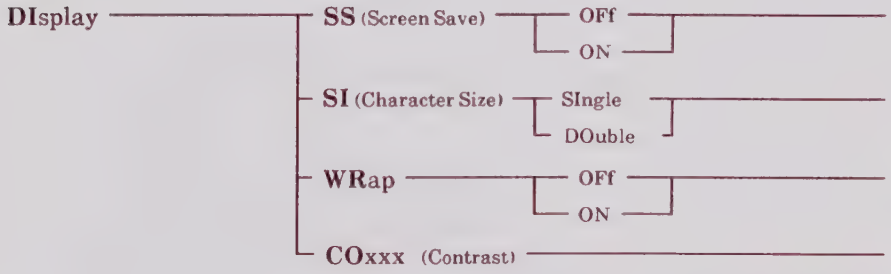
Function **JU** : External reference temperature.

x Junction temperature in °C

MEmory ———————————————

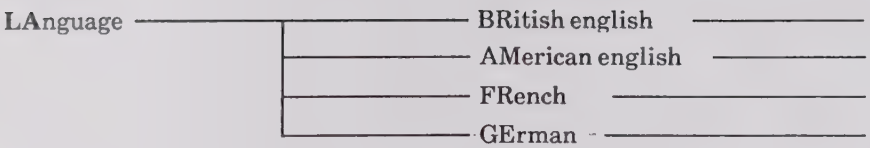
Function **ME** : Reset memory to initial switch-on state; the data base values will return to their default values.

Note : Any remote device will lose command.



Function **DI** : Display Parameters

SS	Screen save on/off
SI	Character Size
	SI Single-size characters
	DO Double-size characters
WR	Wrap facility on/off
COxxx	Contrast setting- adjusts the contrast of the display between 0 and 100%, to optimise readability for user's angle of view.

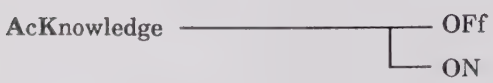


Function **LA** : Language selection

BR	British English
AM	American English
FR	French
GE	German

2.2 ACKNOWLEDGE COMMAND

COMMAND TREE STRUCTURE:



Command mnemonics for the Acknowledge Command Tree are as follows:

Function **AK** : When Acknowledge is ON, the SI3535 will send an "E 000" to a GPIB controlling device whenever a command has been completed. Some commands, such as queries send an output to the controlling device, which acts as an acknowledge; in these cases (or when a command invokes an error response) the E 000 is not sent. This command applies to the GPIB only.

2.3 CHANNEL DEFINITION COMMANDS

Before a channel or channel set may be used by a task, it must first have been defined using the Channel Definition commands. The channel commands associated with each of the following input/output channel types are given in the following sections.

CHANNEL SETS

Channels sets referred to throughout this manual are defined in different categories; the valid formats for these category sets are as follows:

a) 'Single channel' set: specifies the single channel number required.

eg **CH2**

b) 'Multi-channel' set: specifies each channel number separated by either a comma (,) or an ampersand (&). The list of channel numbers is only limited by the total input line length. Channels need not be specified in ascending order.

eg **CH1,35,6,11,4&15&23**

c) 'Range of channels' set: specifies the 'start' channel number and the 'end' channel number separated by a hyphen (-).

eg **CH40-59**

the range may also be specified in descending order.

eg **CH59-40**

or mixed with 'single' or 'multi' channels.

eg **CH1,2,10-20,5**

d) When the SI3535 is controlling one or more slave units it may be required to define a channel set that spans more than one unit. A delimiting character must be used to separate the channels of each unit, ie. channels 1 to 200 of master SI3535 unit channels, 201 to 400, and 401 to 600 for SI3535 slave units.

eg **CH101-200,201-300**

Where channels 101 to 200 are master SI3535 unit channels and 201 to 300 are slave SI3535 channels.

Channels may be entered in any numerical order, and will be scanned in the sequence in which they are entered.

CHANNEL SYNTAX

The following general rules apply when defining the channel set up.

a) The first entry in each new input string, when setting up, querying or changing channel data must always be the channel mnemonic **CH** and the channel number, or channel set.

eg **CH** n-m <**SE** arg><**CO** arg>C/R

b) Channels may be specified in groups, however all channels in a group must be of the same general type (as described in previous section 2.1).

eg all strain gauges, all thermocouples, all digital outputs etc.

c) All channels specified in any one channel definition must be of the same type:

ie either analog inputs,
or event/status input,
or digital input/output

A mixture of types is not permitted.

d) Most of the channel set up parameters are not essential commands. If not defined by the user they will assume their default values. However, the channel mnemonic and channel set must be defined together with the sensor type for the channels.

eg **CH**n **SE**mmm C/R

COMMAND TREE STRUCTURES

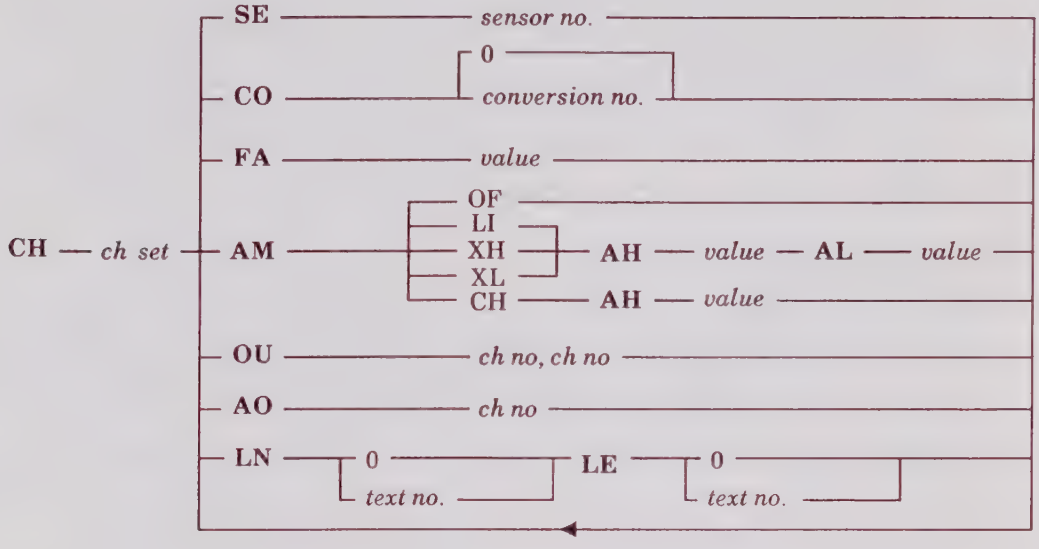
The Command Tree Structures are in the following order :

1. Analog Input Channel
2. Analog Output Channel
3. Decimal-Type Digital Input Channel
4. Status-Type Digital Input Channel
5. Event-Type Digital Input Channel
6. Counter-Timer Digital Input Channel

CHANNEL DEFINITION :

2.3.1 ANALOG INPUT CHANNEL

COMMAND TREE STRUCTURE:



The command tree shows the channel command associated with the definition of the SI3535 Analog Input Channels. The channel mnemonic, **CH**, and channel set must always be the first character entered in the command string, however, all other commands and related arguments are options, and can be entered in any order (or omitted);

CHannel —— *ch set* ——

Function **CH** : The channel mnemonic or channel set numbers affected.

Value ch set : The channel numbers. eg **CH**n, **CH**n-m

SEnsor —— *n* ——

Function **SE**: Selects measurement type to specify type of transducer.

Value *n* : The sensor number; a 3-digit number defining the measurement type, in the following ranges:

110 to 115 Voltage DC measurement
120 to 125 Voltage AC measurement *continued →*

ranges continued...

210 to 279 Resistance measurement
310 to 385 Thermocouple measurement
410 to 479 Platinum Resistance thermometer measurement
510 to 514 Current DC
520 to 525 Current AC
600 to 695 Strain gauges

eg **CHn SEmmm C/R**

Note: See Appendix 2 : Sensor Number Table, which contains more detailed information on sensor numbers.

COnversion ——— *n* ———

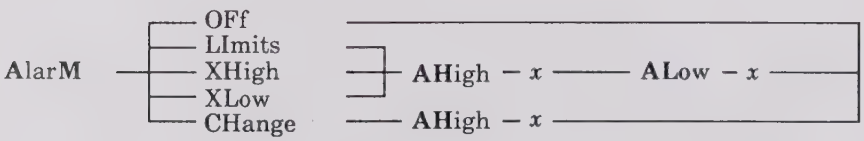
Function **CO**: Defines number of the conversion routine which is to be applied to all logged data. The conversion will be applied to data for all tasks using the channel set. Default value is zero.

Value n: 1-600 conversion routine number, or zero if no conversion to be applied.

FActor ——— *x* ———

Function **FA**: Defines the gauge factor.

Value x: the gauge factor



Function **AM**: Select the alarm type for analog results (default value OF).

OF: Alarm off, but any parameters set up remain in memory.
LI: Specify high and low limits, applicable to all analog SEnsors types except BIT.
XH: Specify threshold value and lower limit of deadband applicable to all analog SEnsor types.
XL: Specify threshold value and upper limit of deadband applicable to all analog SEnsor types.
CH: Specify change size, applicable to all analog SEnsor types.

Function **AH** : Specify upper limit of alarm or deadband, or change size.
Value x : Numeric value defining one of the following: high alarm limit, threshold value, upper limit of deadband, change size.

Function **AL** : Specify lower limit of alarm or deadband.
Value x : Numeric value defining one of the following: low alarm limit, threshold value, lower limit of deadband.

OUtput ——— *n (lo) , n (hi)* ———

Function **OU** : Specifies the two digital output channels associated with 'lo', 'hi' alarms.

Value n digital output channel number.
On low alarm 'lo' goes from logic 'hi' to 'lo'
On high alarm 'hi' goes from logic 'hi' to 'lo'

Analog Output ——— *ch no.* ———

Function **AO** : Specifies the Analog output to be associated with the input channel.

Value ch no : Channel number of the Analog Output Module channel that is to be `attached' to the input channel.

Note : Each Analog Output Module has five output channels, the channel number will be in the range:

1 to 5 if module is fitted in slot 4
21 to 25 if module is fitted in slot 5
181 to 185 if module is fitted in slot 13.

LN ——— *n* ——— **LE** ——— *n* ———

Function **LN** : Text to be output on normal result, specify text number, (0 for off).

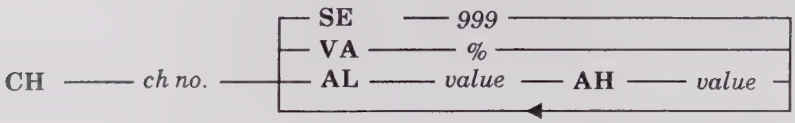
Function **LE** : Text to be output on alarm result specify text number, (0 for off).

Value n : 1-600 text number for text to be output with logged data or alarm data.

CHANNEL DEFINITION :

2.3.2 ANALOG OUTPUT CHANNEL

COMMAND TREE STRUCTURE:



The command tree shows the channel commands associated with the definition of the Analog channels (ie. the output channel associated with a SI3535 input channel).

The channel mnemonic, **CH** and output channel number must always be entered, however the other commands and their arguments are optional and may be entered in any order or omitted.

CHannel ——— *ch no.* ———

Function **CH** : Channel mnemonics.

Value ch no : Channel number.

SEnsor ——— *999* ———

Function **SE** : Selects measurement type to specify type of transducer.

Value 999 : Sensor number for analog output channel.

VAlue ——— *%* ———

Function **VA** : Specifies the set value (percentage) of the output required.

Value % : percentage value (0 to 100%) to be used to obtain a fixed output in the range:

4 to 20 mA if current is selected
0 to 10 V if voltage is selected

where 0% = 4mA or 0 V, 100% = 20 mA or 10 V.

Note : When a **VA**lue has been specified, this overrides any measured value obtained using the 'scale' facility.

ALow ——— *n* ——— **AHigh** ——— *n* ———

Function **AL**: Defines the minimum measurement value, (ie. the value for 0% current or voltage output).

Function **AH**: Defines the maximum measurement value, (ie. the value for 100% current or voltage output).

Value n: The value for 100% full scale current or voltage output (Refer to Note 1).

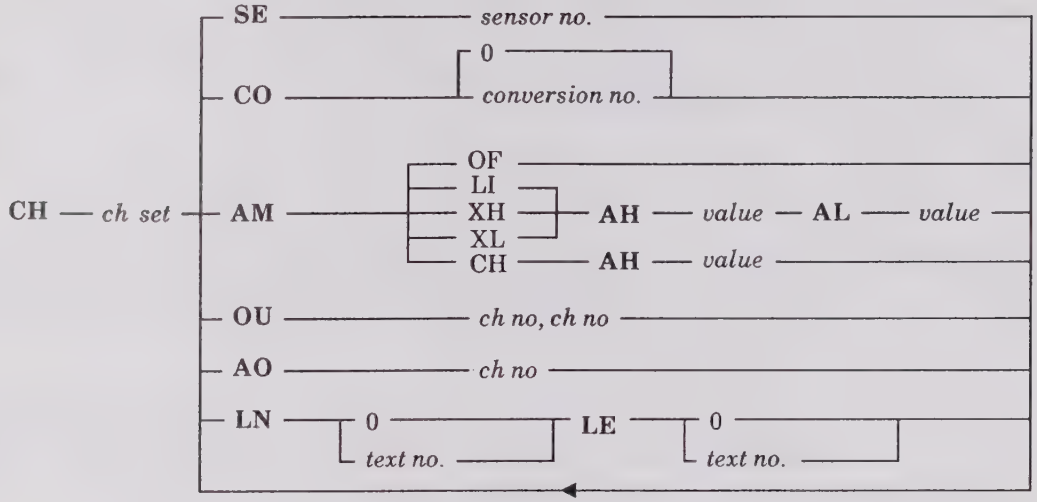
Note 1: The value, n, relates to the measured result for the associated input channel. Enter the numeric value only; engineering units, (eg. µstrain, °C, etc.,) will automatically be assumed depending on the type of sensor connected.

Note 2: Whenever the associated input channel is scanned or monitored, the result obtained causes the analog output channel to produce a proportional current or voltage. The output level will be maintained until updated by the next scan or monitor measurement.

CHANNEL DEFINITION :

2.3.3 DECIMAL - TYPE DIGITAL INPUT CHANNEL

COMMAND TREE STRUCTURE:



The command tree shows the channel command associated with the definition of the SI3535 Decimal Type Digital Input Channels. The channel mnemonic, **CH**, and channel set must always be the first character entered in the command string, however, all other commands and related arguments are options, and can be entered in any order (or omitted);

CHannel —— *ch set* ——————

Function **CH** : The channel mnemonic or channel set numbers affected.

Value ch set : The channel numbers. eg **CH**n, **CH**n-m

SEnsor ———— *n* ————

Function **SE**: Selects measurement type to specify type of transducer.

Value *n* : The sensor number; a 3-digit number defining the measurement type, in the following ranges:

811 to 833 Digital Input

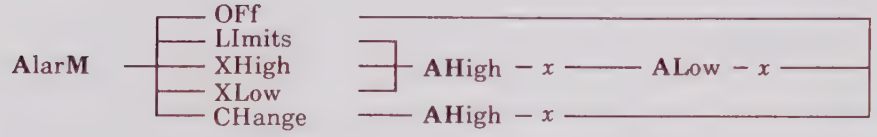
eg **CH**n **SE**mmm C/R

Note : See Appendix 2 : Sensor Number Table, which contains more detailed information on sensor numbers.

COnversion ——— *n* ———

Function **CO** : Defines number of the conversion routine which is to be applied to all logged data. The conversion will be applied to data for all tasks using the channel set. Default value is zero.

Value n : 1-600 conversion routine number, or zero is no conversion to be applied.



Function **AM** : Select the alarm type for analog results (default value OF).

OF : Alarm off, but any parameters set up remain in memory.

LI : Specify high and low limits, applicable to all SEnsors types.

XH : Specify threshold value and lower limit of deadband applicable to all SEnsor types.

XL : Specify threshold value and upper limit of deadband applicable to all SEnsor types.

CH : Specify change size, applicable to all SEnsor types.

Function **AH** : Specify upper limit of alarm or deadband, or change size.
Value x : Numeric value defining one of the following: high alarm limit, threshold value, upper limit of deadband, change size.

Function **AL** : Specify lower limit of alarm or deadband.
Value x : Numeric value defining one of the following: low alarm limit, threshold value, lower limit of deadband.

OUtput ——— *n (lo)* , *n (hi)* ———

Function **OU** : Specifies the two digital output channels associated with 'lo', 'hi' alarms.

Value n : digital output channel number.
On low alarm 'lo' goes from logic 'hi' to 'lo'
On high alarm 'hi' goes from logic 'hi' to 'lo'

Analog Output ——— *ch no.* ———

Function **AO** : Specifies the Analog output to be associated with the input channel.

Value ch no : Channel number of the Analog Output Module channel that is to be 'attached' to the input channel.

Note: Each Analog Output Module has five output channels, the channel number will be in the range:

1 to 5 if module is fitted in slot 4
21 to 25 if module is fitted in slot 5
181 to 185 if module is fitted in slot 13.

LN ——— *n* ——— **LE** ——— *n* ———

Function **LN** : Text to be output on normal result, specify text number, (0 for off).

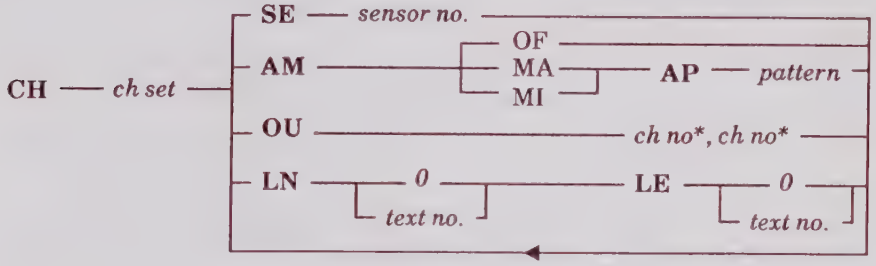
Function **LE** : Text to be output on alarm result specify text number, (0 for off).

Value n : 1-600 text number for text to be output with logged data or alarm data.

CHANNEL DEFINITION :

2.3.4 STATUS - TYPE DIGITAL INPUT CHANNEL

COMMAND TREE STRUCTURE:



*where the channel number is repeated.

The command tree shows the channel commands associated with the definition of the SI3535 Status-type Digital Input Channels. The channel mnemonic, **CH**, and the channel set must always be the first characters in the command string, however, all other command and related arguments are optional and may be entered in any order (or omitted).

CHannel ——— *ch set* ———

Function **CH** : The channel or channel set numbers affected.

Value ch set : The channel numbers.

SEnsor ——— *n* ———

Function **SE** : Selects input measurement type.

Value n : The sensor number (3-digit number), defining the measurement type, in the range:

701 to 708 BIT type status input - Event/Status module
711 to 715 BCD Input - Event/Status module
721 to 725 Inverted BCD input - Event/Status module
801 to 808 BIT type status number - Digital I/O module

Note : Appendix 2, Sensor Number Table contains more detailed information on sensor numbers.

Function **AM** : Select alarm type. Set up pattern matching alarms for status inputs. The available options are outlined below:

OF Alarm off but any parameters set up in memory remain.

MA Select alarm on pattern match, specify bit pattern with which input bit pattern should be compared for match (applicable to BIT type sensors only).

MI Select alarm on pattern mismatch, specify bit pattern with which input bit pattern should be compared for mismatch (applicable to Bit-type sensor only).

Function **AP** : Set up bit pattern for alarm checking.

Value x : 1-8 bits: Alarm pattern against which the input bit pattern will be compared. Channels which are to be ignored for alarm checking purposes should be masked by entering * in the relevant position in the pattern, the corresponding channel is output normally but disregarded by the alarm detector, eg CH41 AP**010 where channels 41 to 45 have been defined as a channel group during sensor definition.

OUtput Alarm —— *n, n* ——————————

Function **OU** :Specify the Digital output channel to be used to match/mismatch alarms.

Value *n* : The Digital output channel number. The channel number should be repeated, separated by a comma.

LN ———— *n* ———— **LE** ———— *n* ————

Function **LN** : Text to be output on alarm result specify text number (default 0 for off).

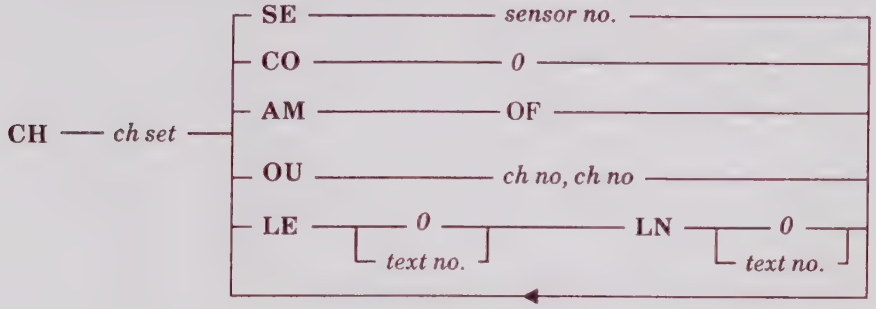
Function **LE** : Text to be output on alarm result specify text number (default 0 for off).

Value *n* : 1-600: text number for text to be output with logged data or alarm data, or events.

CHANNEL DEFINITION :

2.3.5 EVENT - TYPE DIGITAL INPUT CHANNEL

COMMAND TREE STRUCTURE :



The command tree shows the channel commands associated with the definition of SI3535 Event-Type Digital input channels. The channel mnemonic **CH**, and channel set must always be the first characters in the command string, however, all other commands and related arguments are optional and may be entered in any order (or omitted).

CHannel ——— *ch set* ———

Function **CH** : The channel mnemonic.

Value ch set : The channel numbers.

SEnsor ——— *n* ———

Selects **SE** : Selects measurement type.

Value *n* : The sensor number (3-digit number), defining the measurement type, in the range:

740 to 742	Totalize events
750 to 752	Count events
760 to 762	Event capture

Note : Appendix 2, Sensor Number Table, contains more detailed information on sensor numbers.

COnversion ——— *0* ———

Function **CO** : Define conversion number as zero (ie. no conversion applied).

Value 0 : No conversion applied.

AlarM ——— OF ———

Function **AM**: Ensure that alarm monitoring is off.

Value OF : No alarm monitoring.

OUtput ——— *ch no, ch no* ———

Function **OU** : Specify the Digital Output channel(s) to be associated with the event task.

Value ch no : The Digital Output channel number.

LN ——— *n* ——— **LE** ——— *n* ———

Function **LN** : Text to be output with hi-to-lo event, specify text number, (default 0 for off).

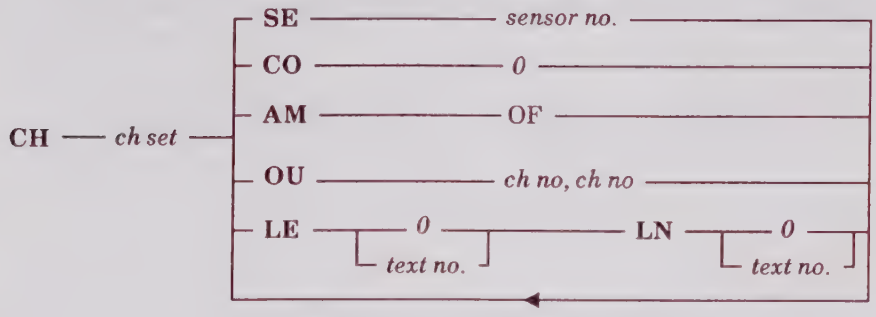
Function **LE** : Text to be output with lo-to-hi event, specify text number, (default 0 for off).

Value *n* : 1-600: text number for text to be output with logged data or alarm data, or events.

CHANNEL DEFINITION :

2.3.6 COUNTER-TIMER DIGITAL INPUT CHANNEL

COMMAND TREE STRUCTURE :



The command tree shows the channel commands associated with the definition of SI3535 Counter-timer input channels. The channel mnemonic **CH**, and channel set must always be the first characters in the command string, however, all other commands and related arguments are optional and may be entered in any order (or omitted).

CHannel ——— *ch set* ———

Function **CH** : The channel mnemonic.

Value ch set : The channel numbers.

SEnsor ——— *n* ———

Selects **SE** : Selects measurement type.

Value *n* : The sensor number (3-digit number), defining the measurement type, in the range:

900 to 904	Frequency
910	Totalize
920 to 925	Period
930	Interval
940	Counter Timer function off.

Note : Appendix 2, Sensor Number Table, contains more detailed information on sensor numbers.

COnversion ——————— *0* ———————

Function **CO** : Define conversion number as zero (ie no conversion applied).

Value 0 : No conversion applied.

AlarM ——————— OF ———————

Function **AM**: Ensure that alarm monitoring is off.

Value OF : No alarm monitoring.

OUtput ——————— *ch no, ch no* ———————

Function **OU** : Specify the Digital Output channel(s) to be associated with the event task.

Value ch no : The Digital Output channel number.

LN ——————— *n* ——————— **LE** ——————— *n* ———

Function **LN** : Text to be output on normal result, specify text number, (default 0 for off).

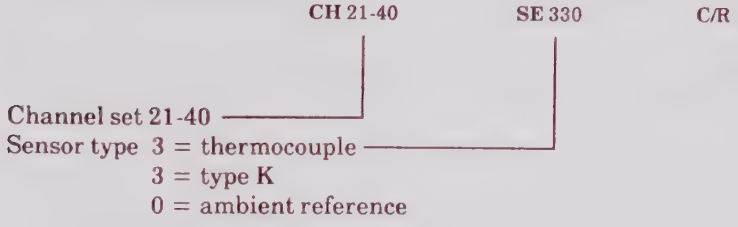
Function **LE** : Text to be output on alarm result specify text number, (default 0 for off).

Value *n* : 1-600: text number for text to be output with logged data or alarm data, or events.

2.3.7 CHANNEL DEFINITION EXAMPLES

The example shown is a typical command string for channel definition. A description of the channel definition is given, followed by the command string broken down into its component parts.

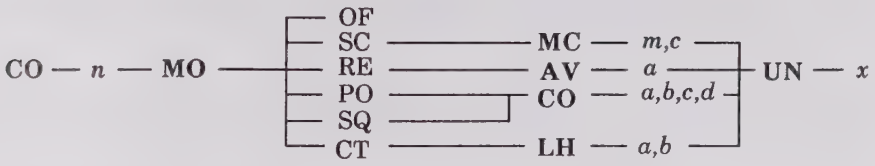
Define channel set 21 to 40 inclusive, as thermocouples sensors type K with ambient thermocouple reference junction. Select an upper alarm limit of 30°C and a lower limit of 20°C.



CH21-40 SE330 AMLI AH30 AL20 C/R

2.4 CONVERSION DEFINITION COMMANDS

COMMAND TREE STRUCTURE

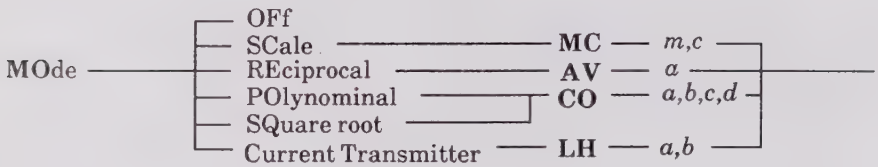


Command mnemonics for Conversion Definition Commands are as follows:

COnversion ——— n ———

Function **CO** : Conversion factor set up.

Value *n* : 1 to 600: conversion routine number, default value 0.



Function **MO** : Specify conversion mode to be used.

OF Deselect the numbered conversion routine but leave parameters in memory.

SC Scaling and offset (mx + c), specify values using MC command.

RE Reciprocal, (a/x) specify value of `a' using AV command Polynomial.

PO Polynomial ($Ax^3 + bx^2 + cx + d$), specify values using CO command.

SQ Square root, ((√(ax + b))*cd), specify values using CO command.

CT Current Transmitter

Function **MC** : Specify scaling values of `m' and `c'.

Value m,c : Scaling `m' and offset `c', decimal values to be used.

Function **AV** : Specify value of `a'.

Value a : Decimal value

Function **CO** : Specify values of a, b, c, d for polynomial, or square root.

Value *a,b,c,d,* : Decimal values

Function **LH** : Specify values for low (4mA) and high (20mA) inputs.

UNits ——— *x*

Function **UN** : Units of result

Value x : Three characters specifying unit of results eg VDC

2.5 TEXT DEFINITION COMMANDS

COMMAND TREE STRUCTURE

LAbel ——— *n* ——— **TX** ——— *text*

Function **LA** : Define text to be output with normal or alarm data. Text is applied to the channels specified by the channel definition commands.

Value *n* : 1 to 600: text number

Function **TX** : Define text string.

Value *text* : 18 characters max. text string.

2.6 TASK DEFINITION COMMANDS

The Tasks that can be set up using the SI3535 TASK commands are of two basic types:

Measurement Tasks, used to scan analog, status or counter timer inputs or event inputs that are to be counted or totalised.

Event Tasks, used for detection of event inputs.

TASK SYNTAX

There are a number of general rules that can be applied when defining a task set-up; these are detailed below:

a) When setting up, changing or querying task data the input string must start with the task mnemonics, **TA**, followed by the task number.

eg **TA** 1

b) When entering the time as a parameter, in the form dd-hh:mm:ss.t, the tenths of the seconds need only be included in the argument if not zero. The value of the seconds should always be specified, but the minutes, hours and days need only be entered if they are significant.

eg enter 10.5 for 10½ seconds
1:00:00 for 1 hour

c) When setting up a task all parameters except TAn, are optional. However, the following parameters must be entered before a task will run successfully:

(i) Task number, **TA**n
(ii) Task type, Measurement **OP** ME or Event **OP** EV
(iii) Task trigger, **TR**, followed by relevant parameter defining trigger type
(iv) Channel set, **CH** ch set, the channel set must also be defined during channel definition.

All other parameters will assume their default values, if SI3535 has been initialized, or alternatively contain any values previously set-up.

COMMAND TREE STRUCTURES

The Command Tree Structures are in the following order :

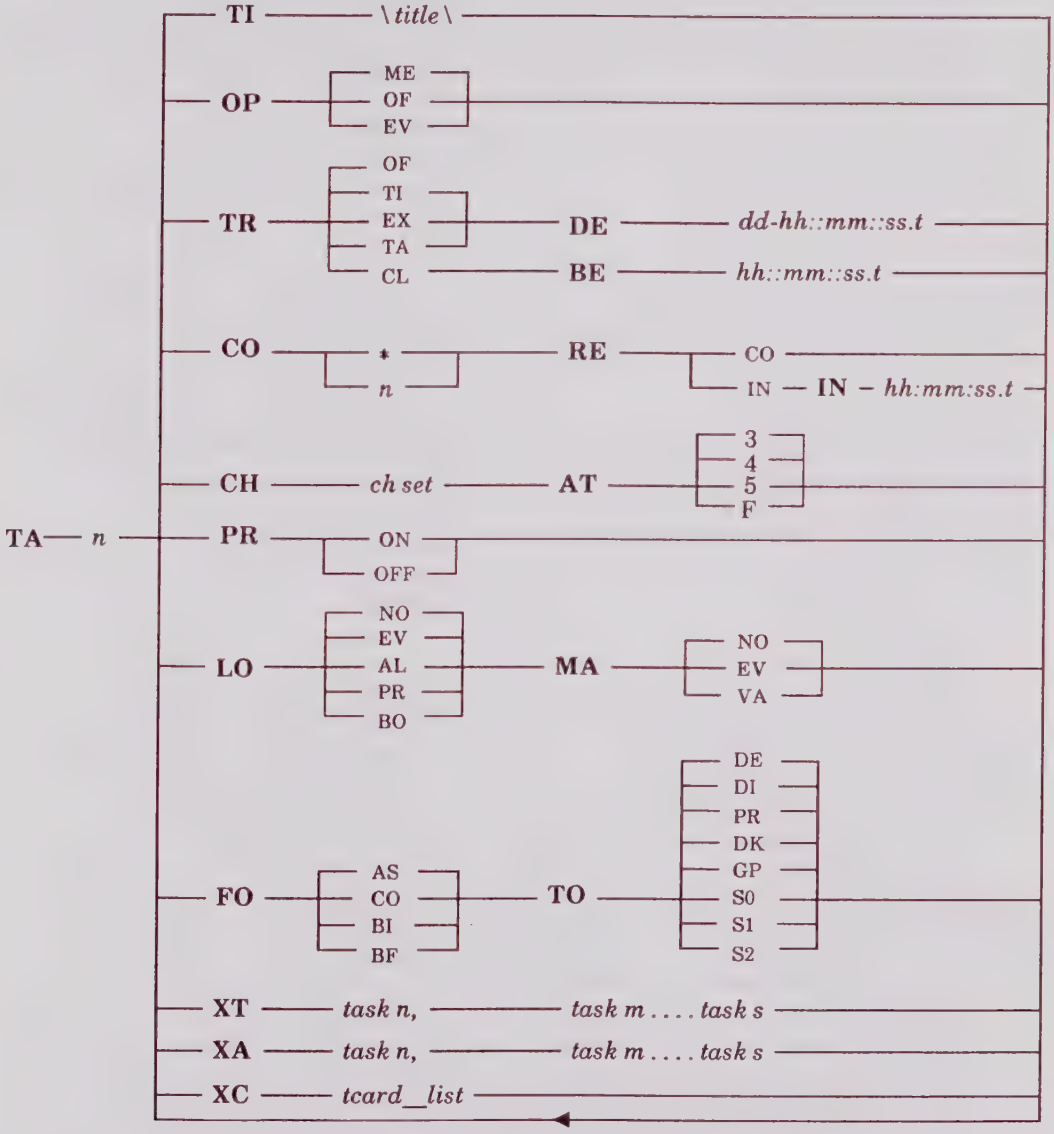
1. Measurement Task

2. Event Task

TASK DEFINITION :

2.6.1 MEASUREMENT TASK

COMMAND TREE STRUCTURE:



The command tree shows the commands associated with the definition of a the SI3535 Scanning Measurement Task. The Task mnemonic, TA, and task number, *n* must always be entered; however all other commands are optional and may be entered in any order (or omitted).

TAsk ——— *n* ———

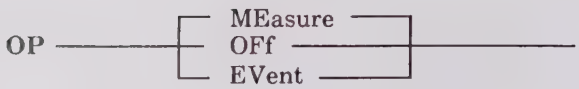
Function **TA** : The Task mnemonic

Value n : The number of the task to be defined.

Function **TI**: Title to be given to the task.

title 18 characters maximum: If the title string contains any spaces it must be preceded by and followed by the delimiting character \.

eg \START UP SEQUENCE\ .

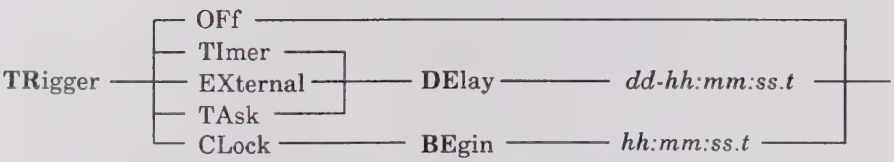


Function **OP**: Define task operation required. Default value if OFf.

OF Will prevent a task from running if it has not yet been triggered. It will have no effect if entered when the task is running, but if the task is subsequently halted it will not restart when re-triggered.

EV Task n is an Event task.

ME The task is a scanning measurement task.



Function **TR**: Task Trigger Default value OFf.

OF Task will not respond to any trigger.
TI Trigger from internal timer, set delay time required.
EX Trigger externally from the execute command, set delay time required.
TA Trigger from another task, set delay time as required.
CL Trigger from internal 24 hour clock at the same time specified by the begin command.

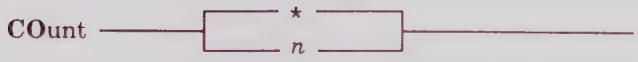
Note 1: The task will automatically retrigger at the same time 24 hours after the last begin time.

DE Specify delay before triggering task. Default value 00-00:00:00.0.

dd-hh:mm:ss.t Delay time required.

BE Specify start time. Default value 00:00:00.0.

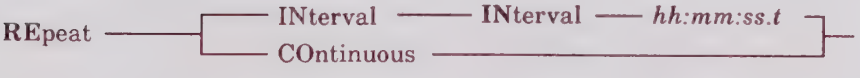
hh:mm:ss.t Start time for trigger from internal clock.



Function **CO**: Define the number of scans required for the measurement/scanning task Default value * (unlimited scanning).

Value * : Unlimited number of scans.

Value n : The number of scans required. The maximum number that can be made is 65535, numbers entered greater than this value will be treated as their modulo remainder, except for zero remainder which will be flagged as an error.



Function **RE**: Define the interval between scans. Default value is continuous, CO.

IN There is to be a defined interval between scans. Interval triggering will initiate measurement scans at the defined interval, subject to task priority. If however the SI3535 buffer becomes completely filled, the system waits for sufficient memory to become available before continuing with the scan.

CO Continuous triggering of measured scans. When continuous triggering is selected the next scan will be triggered when the end of scan marker for the task has been passed to the output interface. This prevents the SI3535 buffer becoming completely filled as the system is slowed down to the speed of the interface.

IN Interval between measurement scans if scanning is not continuous. Default value is 00:00:00.1.

hh:mm:ss.t The measurement delay interval.

CHannel ——— *ch set* ———

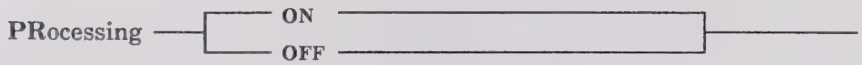
Function **CH** : Define the channel number(s) or channel set to be included in the task.

Value *ch set*: The channels may be specified either as a single channel number, several channel numbers separated by a comma(,) or in the form Channel n to m.

eg TA1 CH5,7,10
TA2 CH11-22,4

Function **AT** : Select the rate at which the measurements are to be made. Four different integration times are available these are set automatically according to the measurement rate selected as follows: (default value is 3)

3	Integration time of 0.625 mS
4	Integration time of 5 mS
5	Integration time of 20 mS
F	Integration time of 80 mS

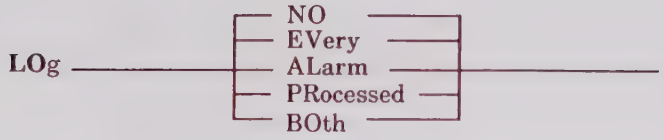


Function **PR** : Select end of scan processing. This will extend the processing capabilities of the task as described in Chapter 4.

When processing is selected the program associated with the task will be performed at the end of each scan. This program can be used to perform sophisticated data reduction on the measured channel data before it is output.

ON	Perform end of scan processing before outputting result data.
OFF	End of scan processing is not performed.

Note 1: Although processing may be off, RUN and Halt markers, and Date change data may be sent to any output device(s) specified in the processing program.

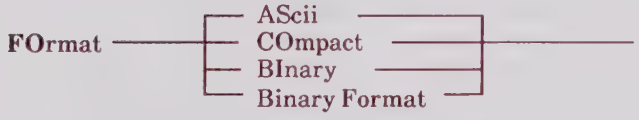


Function **LO** : Define the type or types of data that are to be logged. Default value is NOne.

NO	No data to be logged.
EV	Log all data.
AL	Log only data for channels in alarm.
PR	Log only processed data.
BO	Log both alarm and processed data.

Function **MA** : Specify when header markers are to be output. Default value: NOne.

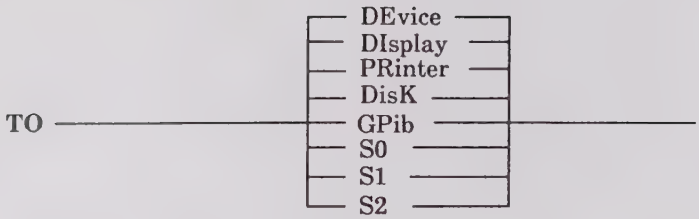
NO	Header markers are not to be output.
EV	Output a header every time the task is actioned.
VA	Output a header only when valid data is logged.



Function **FO** : Specify the format in which the data is to be output to the specified interface. Default value is AS (full format).

AS	Full format: 40 ASCII characters. It includes text messages and titles in addition to full numeric information.
CO	Compact format: 20 ASCII characters. It contains full numeric information, but excludes text.
BI	Coded format: 10 ASCII characters. It contains results in binary format which have been encoded into printable ASCII characters.*
BF	Binary format - Serial Interface and GPIB only.

Note 1 * : If the result is a processed result it may be followed by an additional 10-character string.



Function **TO** : Specify device to which data is to be logged. Default value is DE (null device).

DE	Null device, data is lost
DI	Front panel or local terminal
PR	Front panel printer, (SI3535D) or printer attached to VT220 connected to SI3535F/G
DK	Front panel disk drive SI3535D & G
GP	GPIB interface socket on the GPIB board 35304A
S0	The standard RS232 interface on the controller board
S1	Socket PL1 on the serial interface board 35304B
S2	Socket PL2 on the serial interface board 35304B

XTrigger ——— *task n,* ——— *task m . . . . task s* ———

Function **XT** : To specify which task or tasks are to be cross triggered on an alarm condition.

Value *n...t* : The number or numbers of the tasks that are to be triggered up to a maximum of eight tasks, 1 to 8. (0 = off)

XAbort ——— *task n,* ——— *task m . . . . task s* ———

Function **XA** : Specify the number of tasks that are to be aborted on alarm condition.

Value n...t : The number or numbers of tasks that are to be aborted on an alarm condition. (0 = off)

XChannel ——— *tc_card_list* ———

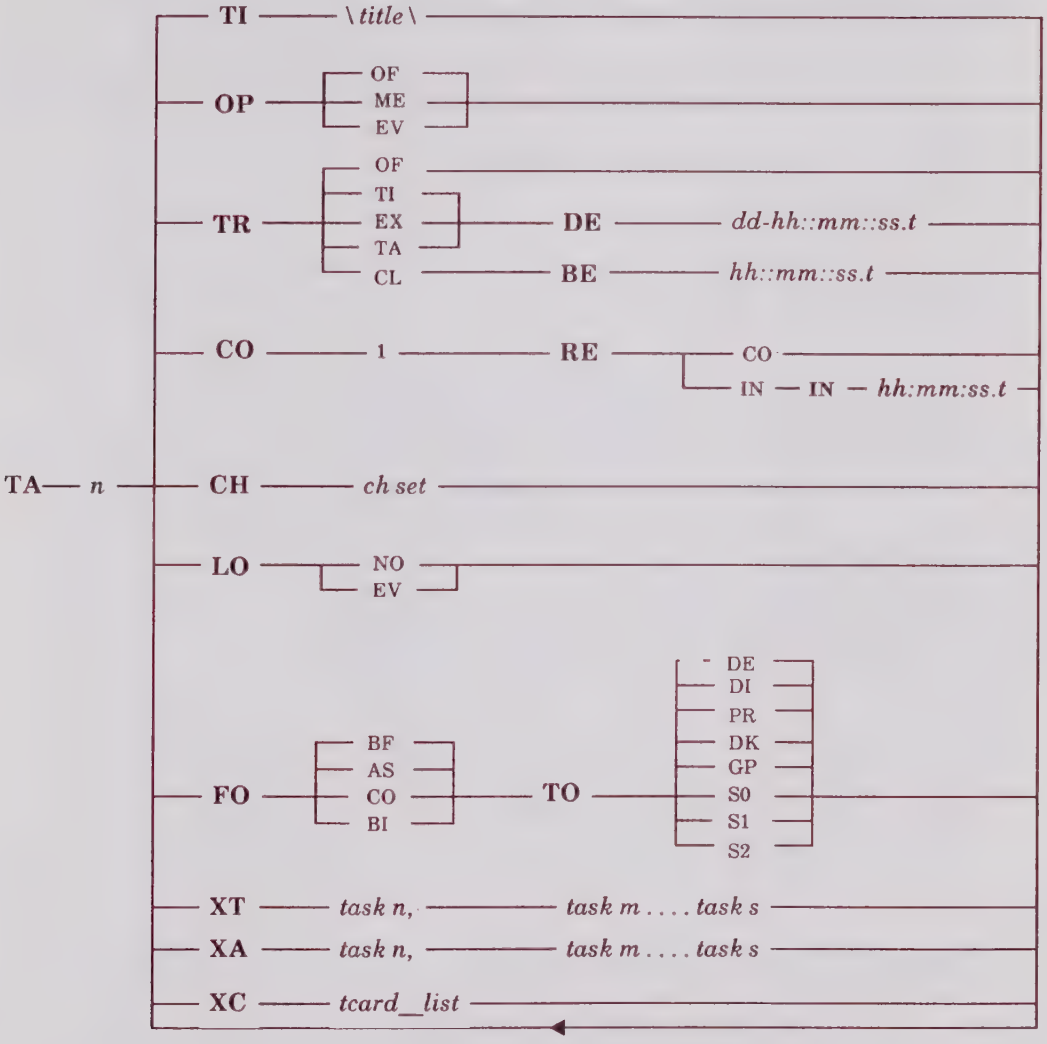
Function **XC** : Arms the given transient cards when an alarm occurs.

Value : Specify the module by the number of its first channel, ie, 21, 41, 61 ....etc

TASK DEFINITION :

2.6.2 EVENT TASK

COMMAND TREE STRUCTURE



The command tree shows the commands associated with the definition of a SI3535 Event Task. The Task Mnemonic, TA, and task number, n, must always be entered; however all other commands are options and may be entered in any order (or omitted).

TAsk ——— *n* ———

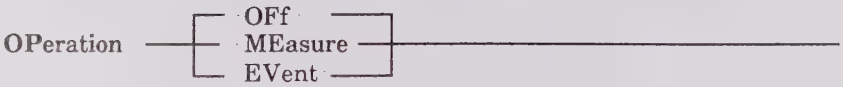
Function **TA** : The Task mnemonic

Value n : The number of the task to be defined.

Function **TI**: Title to be given to the task.

title 18 characters maximum: If the title string contains any spaces it must be preceded by and followed by the delimiting character \.

eg \START UP SEQUENCE\

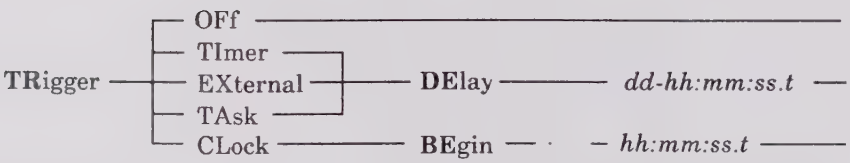


Function **OP**: Define task operation required. Default value if OFf.

OF Will prevent a task from running if it has not yet been triggered. It will have no effect if entered when the task is running, but if the task is subsequently halted it will not restart when re-triggered.

EV Task n is an Event task.

ME The task is a scanning measurement task.



Function **TR**: Task Trigger Default value OFf.

OF Task will not respond to any trigger.
TI Trigger from internal timer, set delay time required.
EX Trigger externally from the execute command, set delay time required.
TA Trigger from another task, set delay time as required.
CL Trigger from internal 24 hour clock at the same time specified by the begin command.

Note 1: The task will automatically retrigger at the same time 24 hours after the last begin time.

COunt ——— *1* ———

Function **CO**: Define the number of scans required for an event task, this should always be defined as 1. The default value is *, continuous.

Value *1*: For an event task the scan count should always be 1.

Function **RE**: Define the interval between scans. Default value is continuous, CO.

IN There is to be a defined interval between scans Interval triggering will initiate measurement scans at the defined interval, subject to task priority. If however the SI3535 buffer becomes completely filled, the system waits for sufficient memory to become available before continuing with the scan.

CO Continuous triggering of measured scans. When continuous triggering is selected the next scan will be triggered when the end of scan marker for the task has been passed to the output interface. This prevents the SI3535 buffer becoming completely filled as the system is slowed down to the speed of the interface.

IN Interval between measurement scans if scanning is not continuous. Default value is 00:00:00.1.

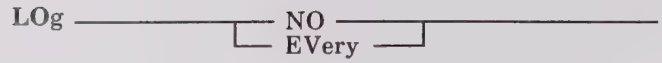
hh:mm:ss.t The measurement delay interval.

CHannel ——— *ch set* ———

Function **CH**: Define the channel number(s) or channel set to be included in the task.

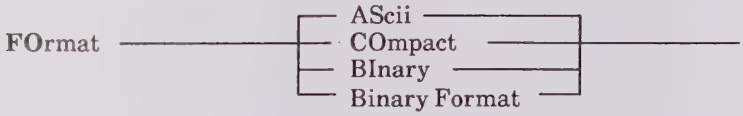
Value *ch set*: The channels may be specified either as a single channel number, several channel numbers separated by a comma(,) or in the form Channel n to m.

eg TA1 CH5,7,10
TA2 CH11-22,4



Function **LO** : Define the type or types of data that are to be logged. Default value is NOne.

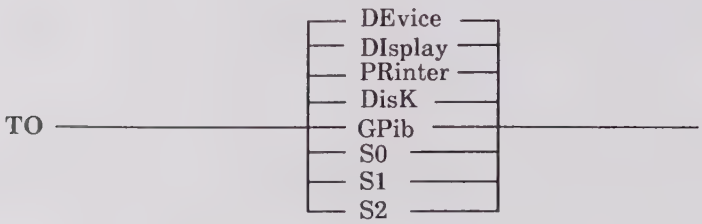
NO	No data to be logged.
EV	Log all data.



Function **FO** : Specify the format in which the data is to be output to the specified interface. Default value is AS (full format).

AS	Full format: 40 ASCII characters. It includes text messages and titles in addition to full numeric information.
CO	Compact format: 20 ASCII characters. It contains full numeric information, but excludes text.
BI	Coded format: 10 ASCII characters. It contains results in binary format which have been encoded into printable ASCII characters.*
BF	Binary format - Serial Interface and GPIB only.

Note 1 * : If the result is a processed result it may be followed by an additional 10-character string.



Function **TO** : Specify device to which data is to be logged. Default value is DE (null device).

DE	Null device, data is lost
DI	Front panel or local terminal
PR	Front panel printer, (SI3535D) or printer attached to VT220 connected to SI3535F/G
DK	Front panel disk drive SI3535D & G
GP	GPIB interface socket on the GPIB board 35304A
S0	The standard RS232 interface on the controller board
S1	Socket PL1 on the serial interface board 35304B
S2	Socket PL2 on the serial interface board 35304B

XTrigger ——— *task n,* ——— *task m . . . . task s* ———

Function **XT** : To specify which task or tasks are to be cross triggered on an alarm condition.

Value *n...t* : The number or numbers of the tasks that are to be triggered up to a maximum of eight tasks, 1 to 8. (0 = off)

XAbort ——— *task n,* ——— *task m . . . . task s* ———

Function **XA** : Specify the number of tasks that are to be aborted on alarm condition.

Value n...t : The number or numbers of tasks that are to be aborted on an alarm condition. (0 = off)

XChannel ——— *tc_card_list* ———

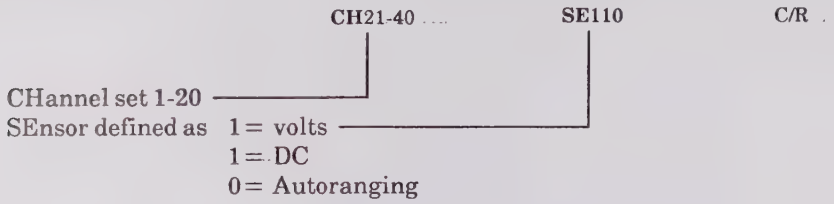
Function **XC** : Arms the given transient cards when an alarm occurs.

Value : Specify the module by the number of its first channel, ie, 21, 41, 61 ....etc

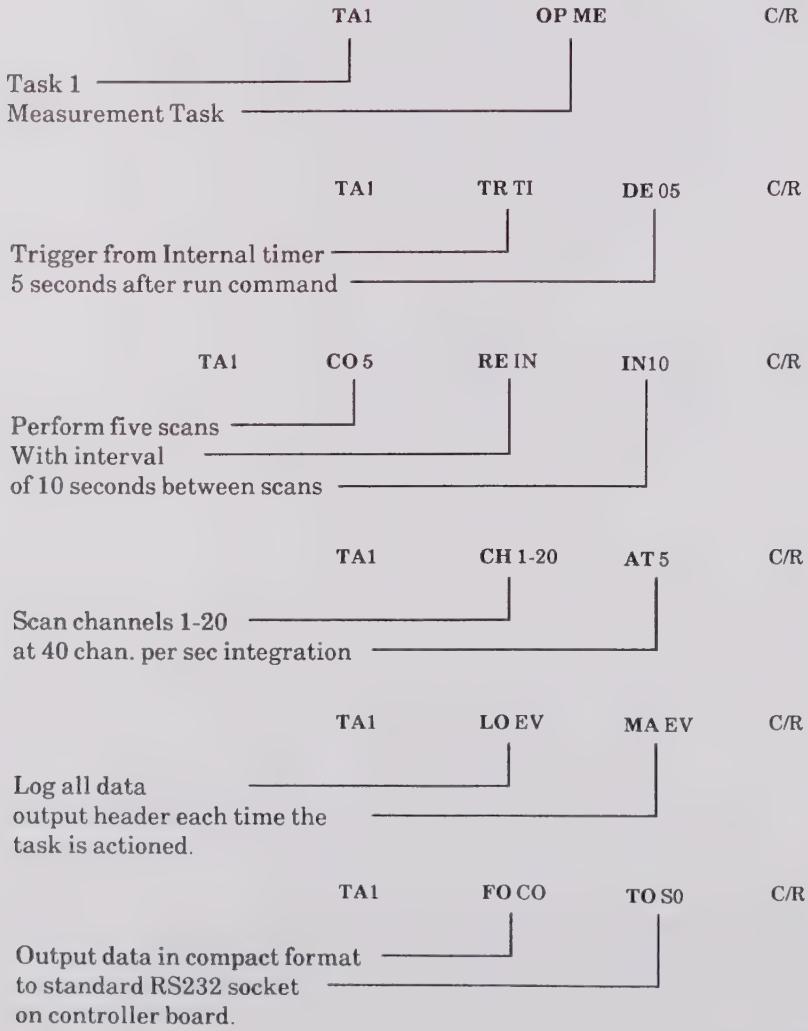
2.6.3 TASK DEFINITION EXAMPLES

Examples of measurement tasks are given below:

a) Define channel set for measurement task as channels 1 to 20 inclusive, measuring volts d.c. and using autoranging.



b) Define a measurement task that is to be triggered from the internal time of five seconds after the **RU**n command has been given. The task is to perform five scans on channels 1 to 20, with a ten second interval between scans. The measurement rate is to be 40 channels per second and all data is to be logged. Data is to be output to the standard RS232 interface in compact ASCII format. A header marker is to be output each time that the task is actioned.



2.7 RENUMBER COMMAND

COMMAND TREE STRUCTURE :

REnumber ——— *Tnnn, Tmmm, pp.*

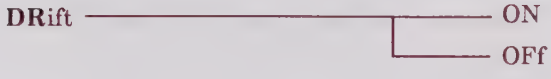
Command mnemonics for the Renumber Command Tree are as follows:

Function **RE** : Renumber program lines.

Value Tnnn Starting at program line number nnn of task T (Old line number).
Tmmm Beginning with the line number of Tmmm (New line number).
pp In increments of pp.

2.8 DRIFT COMMAND

COMMAND TREE STRUCTURE :

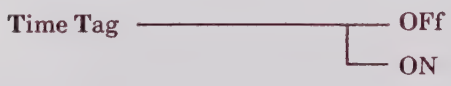


Command mnemonics for the Drift Command Tree are as follows:

Function **DR** : Specify whether drift compensation is required. When ON, periodic measurements of zero are made for each range at appropriate resolutions and the results stored These readings are used as corrections when the active channels are scanned. OFf means that no drift compensation to be applied

2.9 TIME TAG COMMAND

COMMAND TREE STRUCTURE:

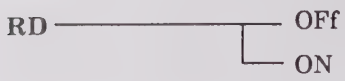


Command mnemonics for the Time Tag Command Tree are as follows:

Function **TG** : When Time Tagging (**TG**) is ON, a time is appended to the channel value. Time tags are only output in Full and Binary Format outputs.

2.10 DIGITAL IO CARD RESET ENABLE/DISABLE COMMAND

COMMAND TREE STRUCTURE:

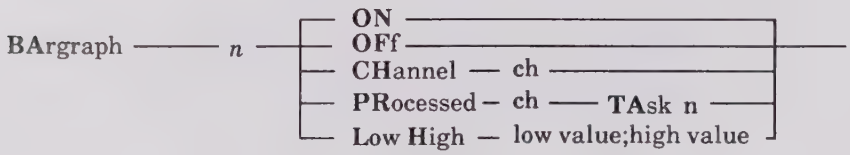


Command mnemonics for the Digital IO Card reset enable/disable Command Tree are as follows:

Function **RD** : This command is used to enable the user to prevent the Digital IO card output latches from being reset every time the logger is halted.
ON enables reset of the Digital IO card latches on HALT.
OFF disables reset of the Digital IO card latches on HALT.
The default is ON and the reset always takes place on power up.

2.11 BARGRAPH COMMAND

COMMAND TREE STRUCTURE :



Function **BA** : Bargraph set up

Value *n* : Bargraph number, 1-10. Up to 10 bargraphs may be switched on but this reduces the area available for text and templates.

ON Selects bargraph to display.

OF Removes bargraph from display.

CH Selects channel source for bargraph.

PR Selects processed channel from Task n.

LH Defines lowest value of bargraph, ie,value at left-hand end and highest value (for right-hand end).

2.12 TEMPLATE COMMAND

COMMAND TREE STRUCTURE :

Function **TE** : Template set up.

Value *n* : Selects one of up to 9 templates. Only one template can be displayed at a time. Each template can contain up to 20 lines, each line consisting of user-defined text and channel markers. These are entered using the Screen Editor.

ON Selects the requested template on to the display. If bargraphs are also switched on, they may overwrite some parts of the template.

OF Removes the template.

DA Selects the source of the data for the template.

MO	Selects the monitor output as the source of data.
TA	Selects task data output as the source of data.
TAn	If the task is selected as the data source, the particular task must be specified.

Editing Templates Remotely

Templates cannot be edited remotely using the screen editor. The templates are accessed in the same way as BASIC lines. Templates start at line 9100 with template 1 and template n at line 9n00. Lines consist of the line number, a single space and then the verbatim text and result markers of the template line.

Entry format for result markers s defined as:

delimiter - type - channel - delimiter

Where the delimiter is one or more asterisks (*)

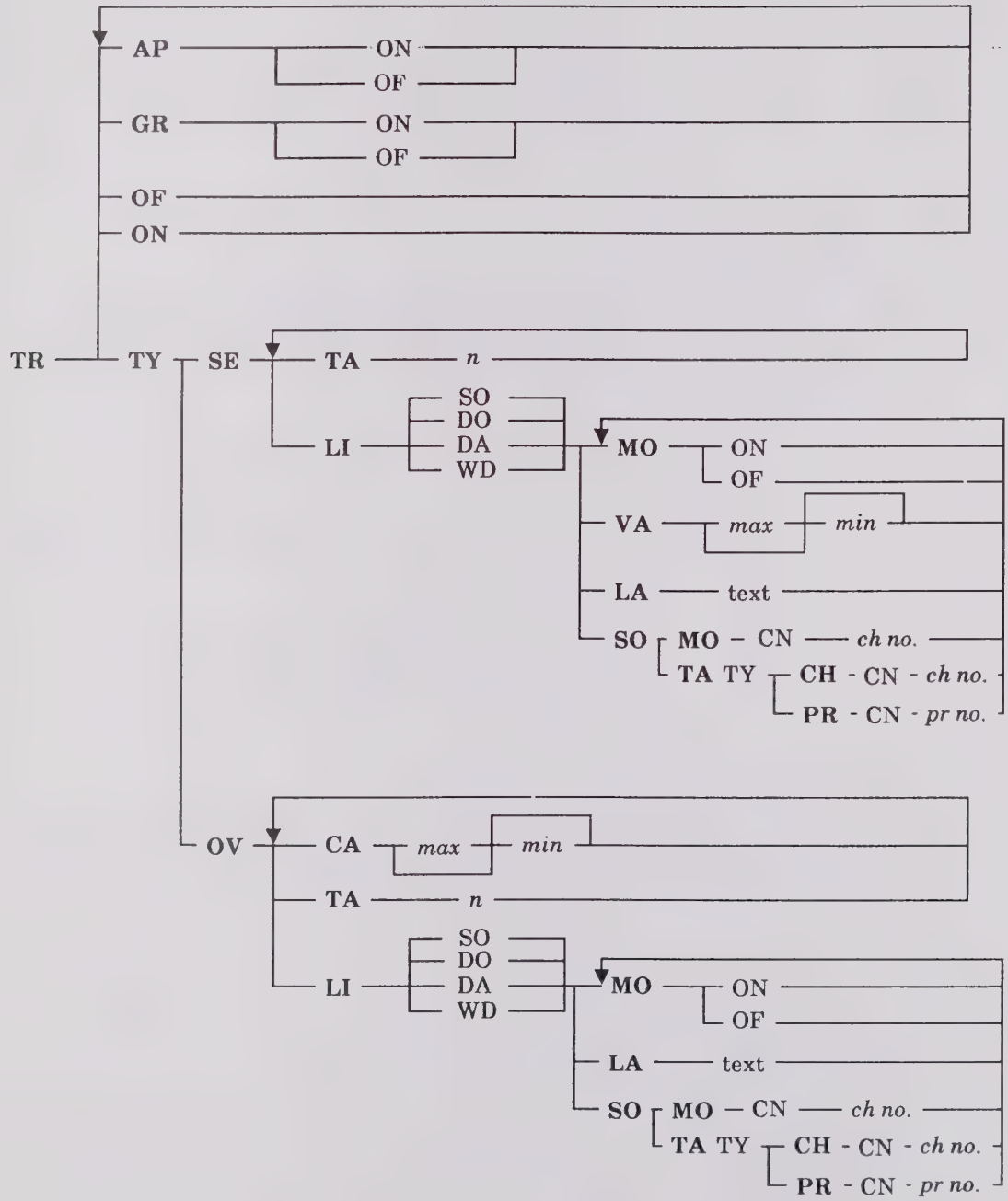
the type is P or C
the channel is a one,two or three digit number

e.g. '*C1*', '*****C1*', or '***C1***'

However, these different input forms expand into the canonical form '***tccc***' on entry. All ASCII characters are valid within a Template with the exception of the asterisk which is reserved for use in template markers.

2.13 TRENDS COMMAND

COMMAND TREE STRUCTURE :

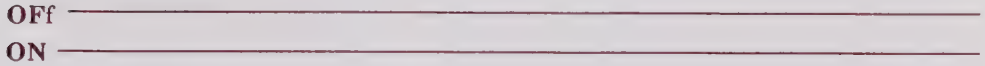


Command mnemonics for the Trends Command Tree are as follows:

Function **TR** : Trend Graph selection and setup.

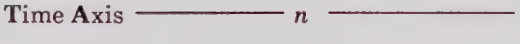
Function **AP**: Specify whether autoprint is required. When ON the SI3535 will automatically screen dump whenever the trend display is full prior to scrolling.

Function **GR**: Specify whether the grid is required. When ON, the grid lines on the time axis will scroll with the display whereas the Y-axis grid will remain fixed. If the grid is OFF, then the graph will have tick markers instead. The grid is available to a maximum of two trends, after which the grid is automatically switched off.



Function **OF**: Specify whether or not to display the trend graph(s). When ON, the
ON graph uses all of the available display area, excluding the menu area and the four bottom lines which are kept for live results or for a template with one line of live results.

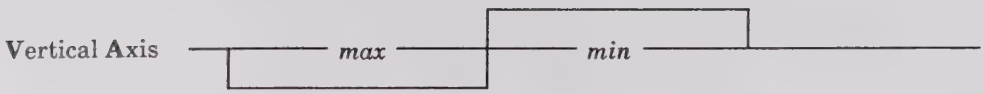
Function **TY**: The format or type in which the graphs are to be displayed. When SEparate is selected, upto four trends can be displayed on separate graphs. When OVerlay is selected upto four trends can be displayed on the same graph. There is no way of combining the two methods; the selection is one or the other.



Function **TA**: The time axis is chosen from the following fixed range:

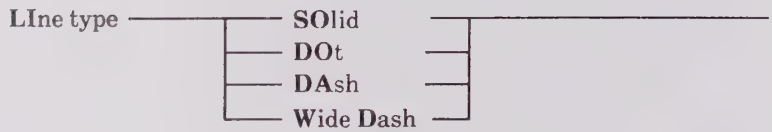
0 = 30 secs	3 = 10 minutes	6 = 12 hours
1 = 1 minute	4 = 30 minutes	7 = 24 hours
2 = 3 minutes	5 = 1 hour	

The time axis (X-axis) is common for all trend graphs if the separate type is selected.

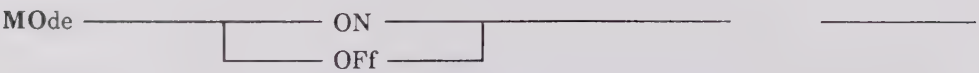


Function **VA** : Specify the maximum and minimum values for the vertical axis. For Separate graphs, the Y-axis may be different for each individual trend graph.

Function **CA** : Specify the maximum and minimum values for the common axis. For a single Overlay graph the Y-axis is common to all trend plots.

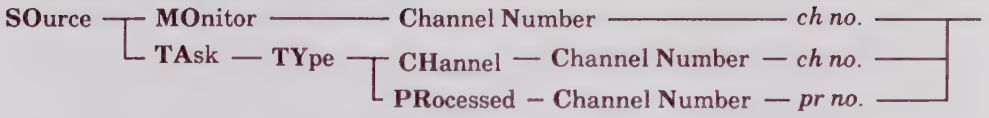


Function **LI** : Specify the line type available to represent the four distinct data sources. When using the overlay graph, at the top of the display there will be a key to show which line type corresponds to which data source.



Function **MO** : Specify whether a trend plot is to be displayed on the graph. If a separate graph type is chosen, when a trend plot is switched Off the remaining graphs are redrawn to take up the remaining space.

Function **LA** : Specify a label to attach to a particular data source. This label will be displayed in the key when overlay type is chosen or next to the Y-axis (vertical axis) when separate trend type is chosen. A maximum of eight characters are permitted.



Function **SO** : Upto a maximum of four trends are allowed, each associated with a single data source. This data source may be any one of the following:

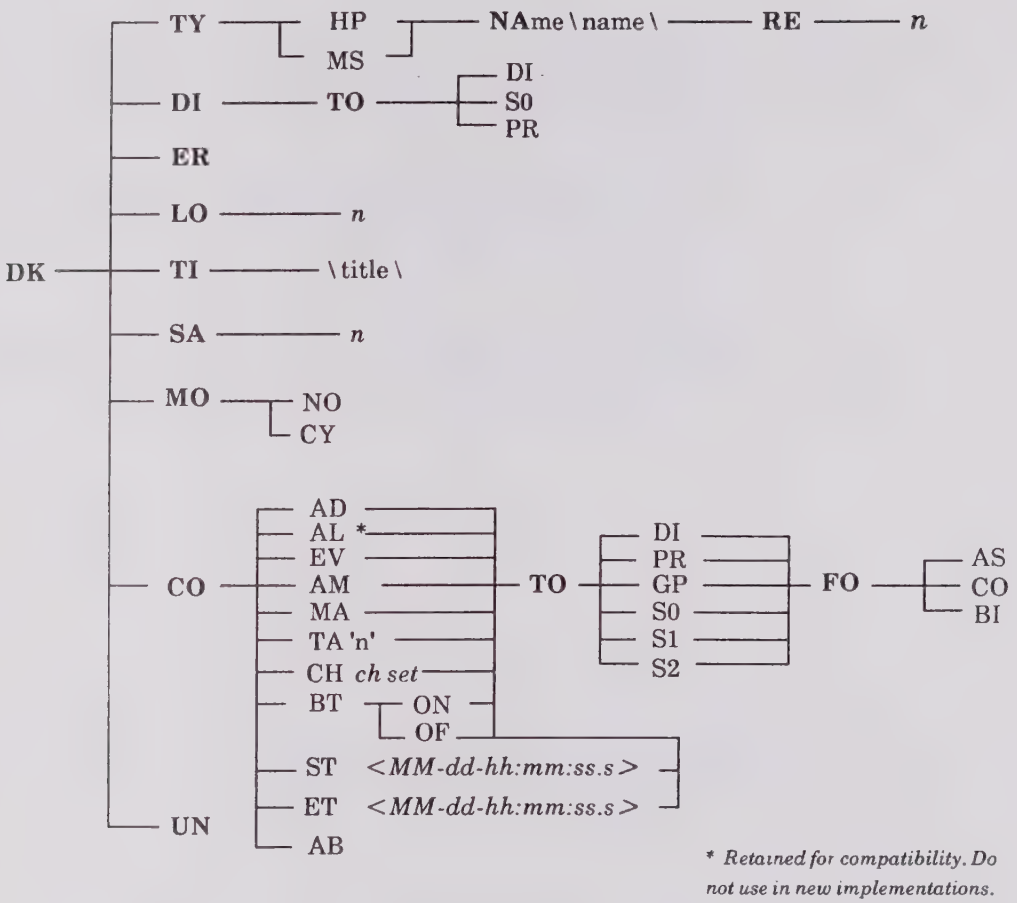
MO CN A single monitored channel. Any channel measured using the monitor facility may be displayed as a trend. This includes any transient capture channel being monitored.

TA TY CH CN A single channel from a specified task scan.

TA TY PR CN A single processed result from a specified task scan.

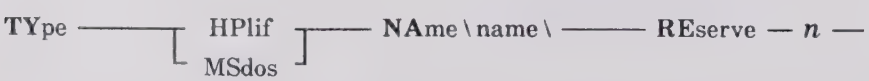
2.14 DISK COMMAND

COMMAND TREE STRUCTURE:



Command mnemonics for the Disk Command Tree are as follows

Function **DK** : Disk command (applicable to SI3535D & G). Use of DK command with SI3535F, results in an error prompt.



Function **TY** : Selects the type of disk format (MSDOS or HP-LIF). Default is MSDOS.

NA Specifies the disk volume name. The delimited string\name\should meet the naming conventions of the selected disk format.

eg MSDOS: up to 11 characters
HP-LIF: up to 6 characters

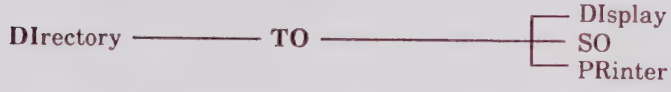
The first character in the delimited string must be an upper case letter. In the remainder of the string, numbers can be used, but there must be no space character.

RE Format the disk, reserving space for the number of programs required.
n 1 to 4: number of program spaces to be reserved. (Only 3 for HP-LIF).

Note: If no volume name has been specified then one will be assigned based on current time:

name M D HH mm

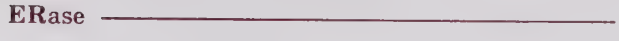
M	= month code: A to L	= Jan to Dec
D	= day code: 1 to 9, A to V	= 1 to 31
HH	= hours code: 00 to 23	= 0 to 23
mm	= minute code: 00 to 59	= 0 to 59



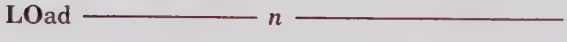
Function **DI**: List program numbers and names on devices specified.

TO Specify device to which directory is to be output:

DI	Front panel display
S0	Standard RS232 on Controller module
PR	Front panel printer

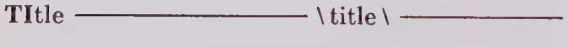


Function **ER**: Erase logged data but not the programs



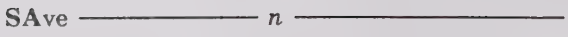
Function **LO**: Load the specified program from disk into memory.

Value *n*: 1 to 4: program to be loaded into memory. (Only 3 for HP-LIF).



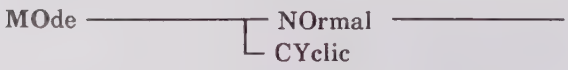
Function **TI**: Specify title to be given to program to be saved.

\title\ Up to 40 character title. The delimiting character \ must be used before and after the character string, eg \TESTPROG.A\.



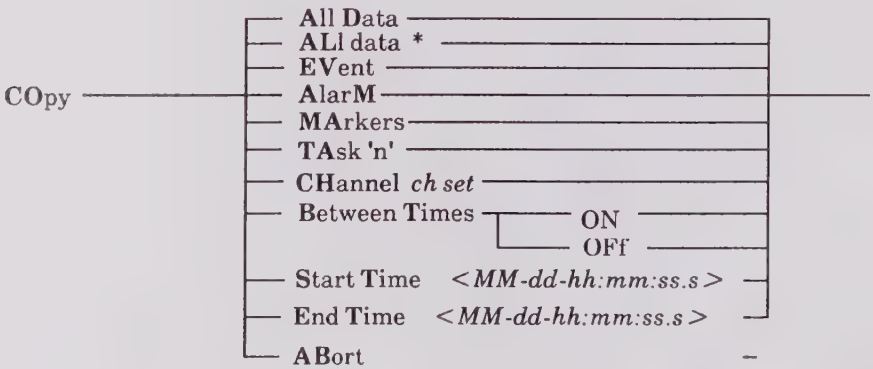
Function **SA** : Save program on disk. For strain gauge channels any initialized value will be saved.

Value n : 1 to 4: the program in memory will be saved with this reference number. (Only 3 for HP-LIF).



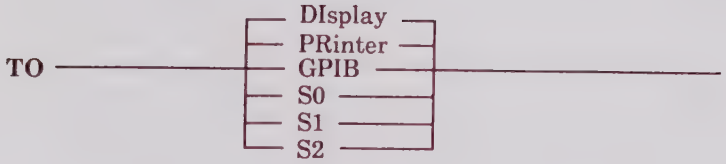
Function **MO** : Specify storage mode on disk:

NO Normal mode: write data to disk until disk is full.
CY Cyclic mode: write data to disk until disk is full, then begin to overwrite with new data, starting where the data is 'oldest'.



Function **CO** : Data specified is copied from the disk to an external output device.

AD Replay All data for all types of results
AL Replay All data (for compatibility with previous issues) * Do not use this command with the new software It is intended for compatibility reasons only.
EV Replay Event data only
AM Replay Alarm data only
MA Replay Markers only
TAn Replay Data from the specified task (n= 1...8 or *)
CH Replay Data from the specified channels (standard channel set (see page 3.9) or *)
BT ON Select results to replay by Time (on or off)
OFf
ST Start Time <MM-dd-hh:mm:ss.s>
ET End Time <MM-dd-hh:mm:ss.s>
AB Stop copying data from disk.



Function **TO** : Specify device to which data from disk is to be copied.

DI	Front panel display or local terminal
PR	Front panel printer
S0	Standard RS232 on Controller module
S1	Socket PL1 on Serial Interface module 35304B
S2	Socket PL2 on Serial Interface module 35304B
GP	GPIB interface socket on GPIB module 35304A.

Function **FO** : Select format in which data is to be output.

AS	Full format: 40 ASCII characters per channel, including text.
CO	Compact format: 20 ASCII characters per channel, without text.
BI	Coded: 10/20 ASCII characters per channel depending on whether the data is processed data.

The format that can be output is conditional on the format in which the record was made:

Record full format:	Copy in full format only
Record compact format:	Copy compact format only
Record coded encoded	Copy output may be any one of the three selections, as below:

a) Binary encoded: the output device must be able to use the code, or if for display or hard copy, must decode to decimal format.

b) Compact format: decoded within the SI3535.

c) Full format: decoded within the SI3535.

Note: When decoding binary coded data for output in either full or compact format, the SI3535 must contain the program and option card under which the data was recorded. The program must be identical in every respect to the original program particularly if the disk contained processed data, otherwise the data will not be decoded correctly.

UNload ————————————

Function **UN** : Prepare the disk for removal. This command allows the user to manually pause the logger, change the disk and then continue running. The facility is provided for both the front panel and remotely controlled versions of the SI3535.

The required command sequence is as follows:

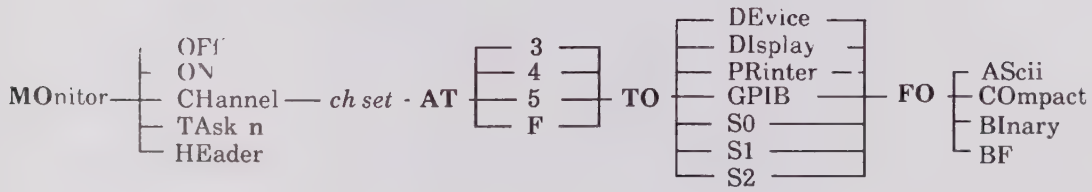
PA Pause the output to disk (and also the printer)

DK UN Following the Pause command, leaves the disk in a state, in which it may be removed and a new one inserted.

GO Resume the output

2.15 MONITOR COMMAND

COMMAND TREE STRUCTURE :



Command mnemonics for the Monitor Command Tree are as follows:

Function **MO** : Set-up the parameters for the monitoring facility; initiate or stop the monitoring of channels, tasks or headers.

Note 1: The monitor command can be set up and initiated when a task is running.

Note 2: Monitoring of the specified channels, task or header will start immediately that the command input string has been entered.

OF Monitoring stops but all parameters remain in a set-up state.

ON Select ON when monitoring parameters have been set-up to start monitoring of channels, tasks or headers.

CH Select monitoring of channels, when the monitor scan is initiated, results are output to device specified by the TO command.

ch set channel numbers to be monitored.

AT Define measurement rate required. Default value is 3.

3	Measure at 500 or 150 channels per second (0.625 mS)
4	Measure at 100 channels per second (5 mS)
5	Measure at 40 channels per second (20 mS)
F	Measure at 10 channels per second (80 mS)

TO Specify device to which data is to be output. Default value is DE.

DE	Null device, output lost
DI	Front panel display or local terminal
PR	Front panel printer (3535D only)
GP	GPIB interface socket on Controller module
S0	Standard RS232 socket on Controller module
S1	Socket PL1 on Serial Interface module 35304B
S2	Socket PL2 on Serial Interface module 35304B

FO Format in which data is to be output .Default value ASCII).

AS	Full format 40 ASCII characters; text is included.
CO	Compact 20 ASCII characters; no text included.
BI	Binary encoded 10 ASCII characters; data values are converted to printable ASCII characters.
BF	Binary Format

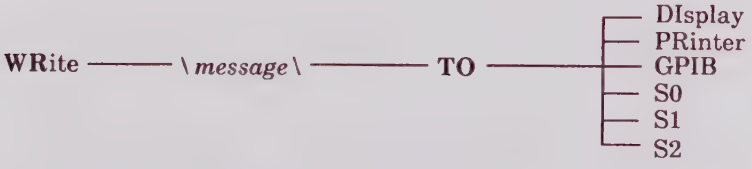
TA Select task monitoring, results are output to device specified by TO command when the monitored task is running.

n specify task to be monitored

HE Select monitoring of headers. Start task marker (STn) ,time of start and end task marker (DTn) are output for each scan

2.16 WRITE MESSAGE COMMAND

COMMAND TREE STRUCTURE :



Function **WR** : Send a message

message 40 characters max. message to be sent to device specified. The delimiting character \ must be used before and after the character string.

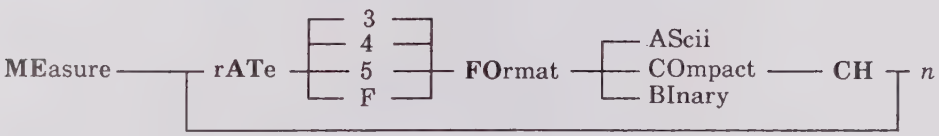
eg \HAVE TAKEN CONTROL\

TO Specify device to which message is to be sent.

DI Front panel display or terminal
PR Front panel printer
GP GPIB interface socket on GPIB interface board 35304A
S0 Standard RS232 socket on controller board
S1 Socket PL1 on serial interface board 35304B
S2 Socket PL2 on serial interface board 35304B

2.17 MEASURE COMMAND

COMMAND TREE STRUCTURE :



Command mnemonics for the Measure Command are as follows:

Function **ME** : An immediate measurement is made on the single channel specified. The response to the measurement is sent to the device which issued the command. The integration will be ignored when measuring on digital channels. A short form of the command (MEn) can be used in which case the CH mnemonic is not required and the integration and format values set up for a previous measure are used.

AT Specify measurement integration. Default value 3

3 Measurement rate 500 or 150 channels per second
4 Measurement rate 100 channels per second
5 Measurement rate 40 channels per second
F Measurement rate 10 channels per second

FO Format in which result is to be output. Default value AS.

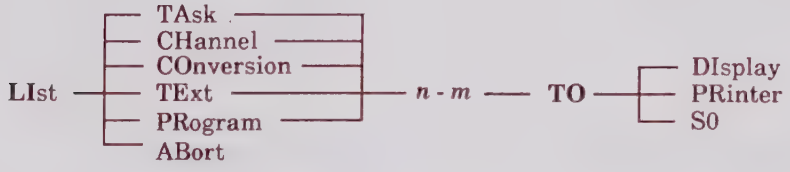
AS Full format 40 ASCII characters; text is included.
CO Compact format 20 ASCII characters; text is not included.
BI Binary encoded 10 ASCII characters; data values are converted to printable ASCII.

CH Specify number to be measured.

n Channel number to be measured.

2.18 LIST COMMAND

COMMAND TREE STRUCTURE :



Command mnemonics for the LIST Command are as follows:

Function **LI** : List the existing set up parameters of the required function on the device specified.

TA List the current set-up parameters for the task or tasks specified.

CH List the current set-up parameters for all channels present in the range specified.

CO List the current conversion set-up for conversion numbers specified.

TE List the text specified by the text number(s).

PR List the program or programs specified.

n-m The value of n and m are dependent on the function specified.

Task:	value in range 1 to 8, a single task may be specified or a group of Tasks, eg 1 or 5
Channel:	value in range 1 to 200, a single channel or group of channels may be specified. Only details of channels actually present will be listed.
Conversion:	value in range 1 to 40, specified singularly as a group eg 3 or 10-15.
Text:	value in range 1 to 40, specified singularly or as a group.
Program:	line numbers for program to be listed, eg 1000-1100.

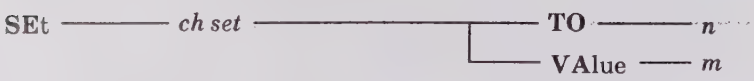
AB Abort the current command.

Function **TO** : Specify device to which output is to be sent.

DI Front panel display (RS232)
PR Front panel printer (SI3535D only)
S0 Standard RS232 socket on controller board

2.19 SET COMMAND

COMMAND TREE STRUCTURE :



Command mnemonics for the Set Command Tree are as follows:

Function **SE** : Set the required bit pattern in a channel set, or the percentage value for an Analog Output Channel.

Value *ch set* : The channel number or channel set

TO Specify the bit pattern required.

n Bit pattern (binary value)

VA Specify the percentage value to be used for the output value.

m Percentage value to be used to obtain a fixed output in the range:

4 to 20mA if current is selected
0 to 10v if voltage is selected

where: 0% = 4mA 0v, 100% = 20mA or 10v

2.20 CONTROL COMMANDS

The SI3535 Control Commands tree structures are defined as follows:

RUn ——————————

Function **RU** : Start tasks: all programmed tasks become operational. The SI3535 waits for the first valid task trigger, then triggers each task.

When the RUN command has been given, the SI3535 tasks can be queried using the Query command, but tasks cannot be modified, and no further tasks may be defined.

HAlt ——————————

Function **HA** : Stop task: All tasks running or waiting for a valid trigger are halted. Task and channel setup data can be entered or modified while the SI3535 is in the HALT state.

PAuse ——————————

Function **PA** : Pause: does not halt the operation of tasks which are running, but suppresses the output of data to the interfaces. The programmed tasks will continue to run but data will be buffered up until the GO command is used to cancel the PAuse if however the buffer becomes full the tasks will be temporarily suspended until data has been output and there is room in the buffer for further data.

GO ——————————

Function **GO** : Continue: Cancels the pause. All programmed tasks are fully operational, buffered data is output to the relevant device.

XT Cross Triggering

Function **XT** : Cross Trigger: sends a trigger to the specified task. Used primarily with tasks which were defined to have external trigger. Works only when instrument is running. XT can be used to re-start a task aborted by XA.

XTrigger n,m....p

n,m...p The number of the task or tasks to be triggered. Up to eight tasks may be included in the command, each separated by a comma(,).

XA Cross Abort

Function **XA** : Cross Abort: abort the specified task or tasks.

XAbortn,m...p

n,m...p The number of task or tasks to be aborted. Up to eight tasks may be included in the command, each separated by a comma(,).

PRint SCreen

Function **PR SC** : produces a hard copy of the display on the printer

2.21 QUERY COMMAND

The function of the query command is to interrogate the SI3535 data base. Queries do not change the state of the data base and therefore unlike commands, may be made at any time (ie. by a device not in full 'LOCAL' or 'REMOTE' control).

With the following exceptions, all parameters may be queried :

The first parameter in an input command string or single command string,

ie. TA task, PA pause, CO conversion, MO monitor, CH channel etc.

Any of the above will result in the error response E003 (ie. no query available).

Querying a Parameter

To query a parameter value the command mnemonic must be entered followed by a question mark (?) which replaces the normal parameter mnemonic or value.

eg. TA1 TR? will query the task trigger defined at present for task 1

Note 1: A space is not permissible before the question mark (?).

The current value for the queried parameter, held in the SI3535 data base, is displayed at the terminal from which the enquiry was made.

For the example above, one of the values: OF, TI, EX, IN or CL would be displayed.

Note 2: The reply to a query will always be returned in the format in which the parameters would normally be entered.

eg. A query for the current date held in SI3535 : IN DA?

Response would be in the form : dd-mm-yy or mm-dd-yy (USA)

It is also possible to query a BASIC line. eg. 1000?

Query Only Commands

Three commands are available which are solely used to query the interface cards present in the SI3535. Their format and functions are as follows:

A) Query interface type: **TY**

CHannel ———— *ch set* ———————————— **TY?**

The SI3535 replies with one of the following:

DI	For a Digital I/O card in the slot for corresponding channels.
AN	An Analog card either reed relay, solid state or HVSS in the slot for corresponding channels.
CO	Counter Timer module in slot for corresponding channels.
DA	Digital to Analog converter module in slot for corresponding channels.
EV	Event/Status module in slot for corresponding channels.
TC	Transient Capture card.
E254	There is no card present in slot or there is a channel conflict, ie. channels specified are on two different cards.

B) Query digital status: **DS**

CHannel ———— *ch set* ———————————— **DS?**

The SI3535 replies with either :

DI	For digital alarm mode, where an alarm is checked for a bit pattern match or mismatch
AN	In all other cases.

C) Query Analog Output module configuration.: **VI?**

CHannel ——— *ch no* ——— **VI?**

The voltage or current drive capability is switch selectable on the Analog Output card, but the status of a given channel may be queried by: **VI?**

The SI3535 replies with either:

VO	Configured for voltage output.
CU	Configured for current output.

D) Query Clock Synchronization: **SY?**

When more than one SCORPIO is being used in a measurement system, it may be desirable that the clocks in the instruments be synchronised. This is queried and adjusted as follows:

SY? This command returns the time in the binary format below,

YYY00000 MMMMYYYY hhhDDDDD mmmmmmhh llssssss lllllll

which is the same format as for binary time tags (Chapt 5, Section 4.2).

SY FA? This command tells the SI3535 that its clock is fast. Rather than decrement time, which could cause time tagged results to move backwards in time, the time will not be decremented, but will stand still for 5 milliseconds. The action routine will read the current time, subtract 5 milliseconds and return in the same format as for the 'SY?' command.

SY SL? This command tells the SI3535 that its clock is slow. The time is incremented by 5 milliseconds, and the new time is returned in the same format as for the 'SY?' command.

Status Query Command

The Status Query command provides a summary of the current status of the SI3535. When the command is sent from the PC to the SI3535, the information is returned as a single ASCII string, the first byte being the Run status, and the last byte the seconds of the time of day. Task status information consists of eight characters: the first character indicates task 1 status, and the remaining tasks follow.

Note: The Version Code is used to differentiate between possible future versions of the SI3535 software.

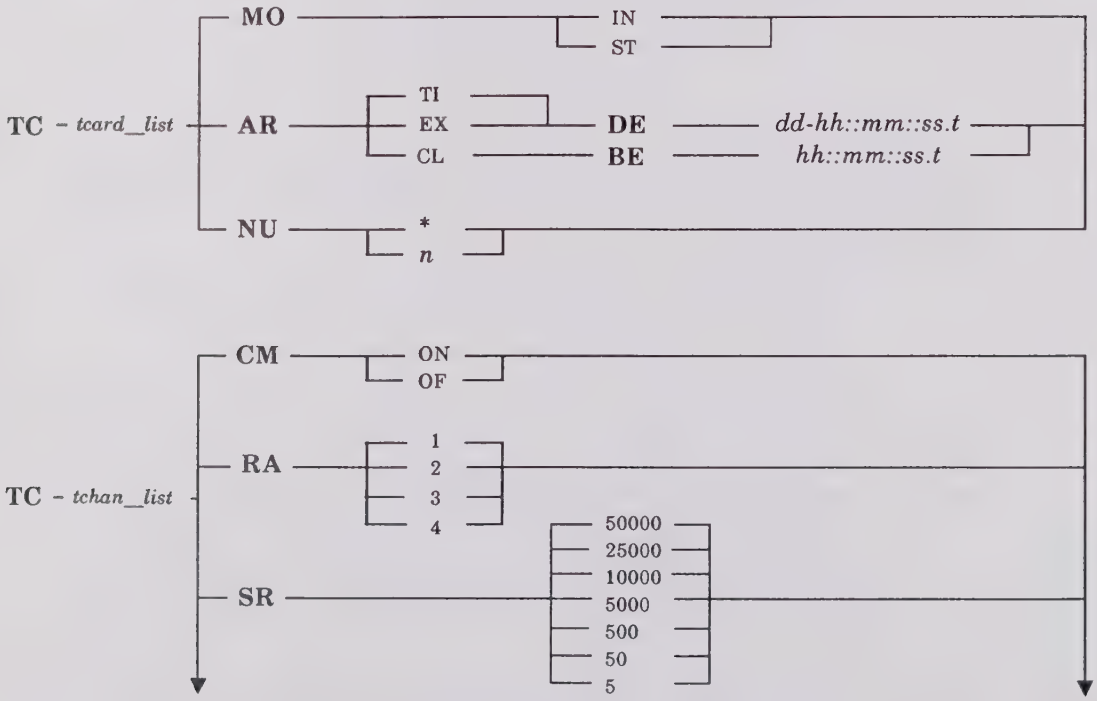
The command to query the status is:

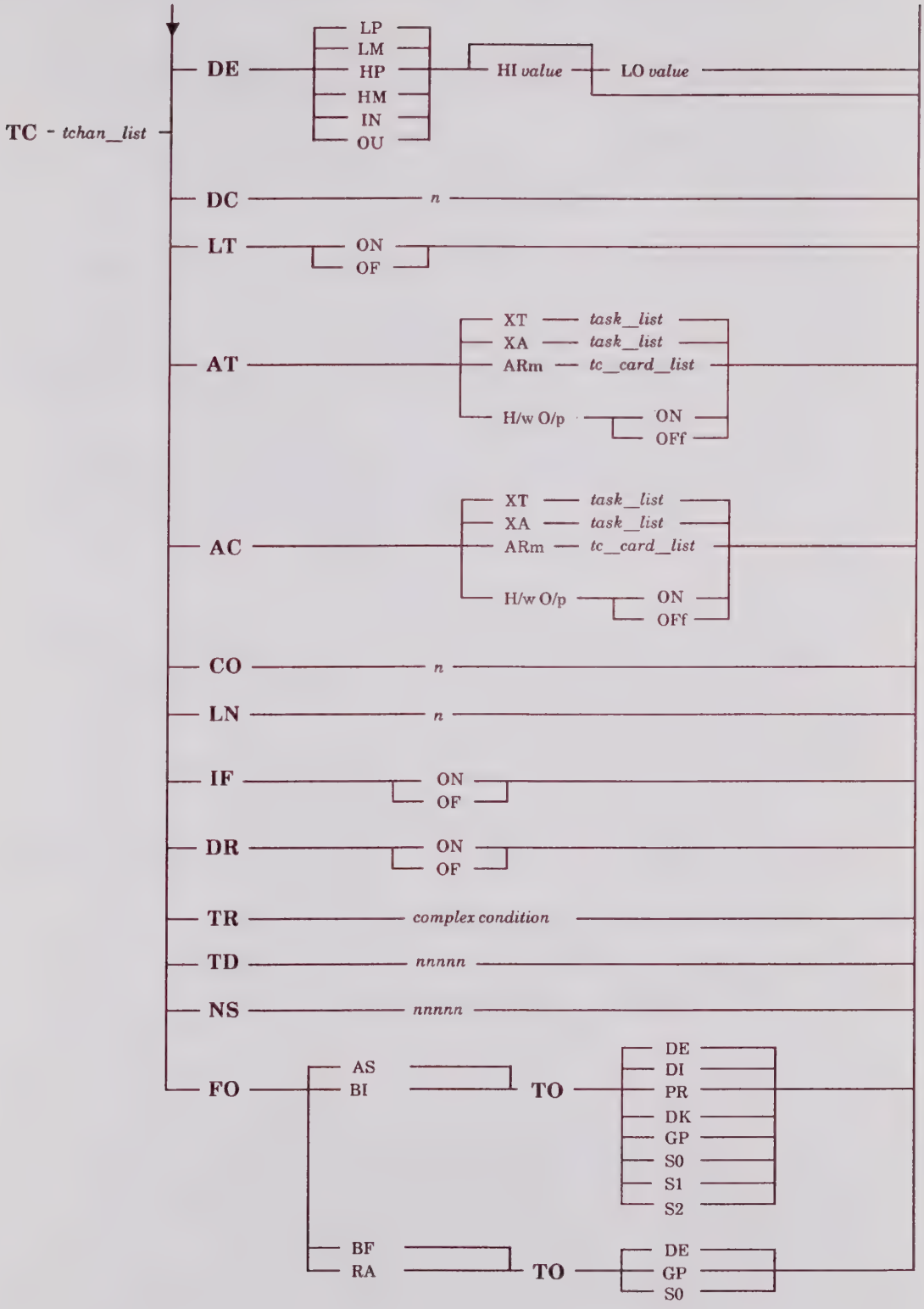
ST? Output will be directed to the command response stream

Refer to table 5.1 in chapter 5 for the format of the returned value.

2.22 TRANSIENT CAPTURE COMMAND

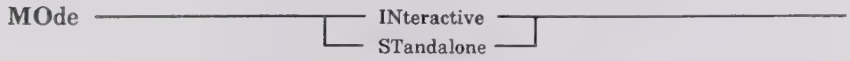
COMMAND TREE STRUCTURE :




TC - tchan_list
DE
LP
LM
HP
HM
IN
OU
HI value
LO value
DC
n
LT
ON
OF
AT
XT
task_list
XA
task_list
ARm
tc_card_list
H/w O/p
ON
OFf
AC
XT
task_list
XA
task_list
ARm
tc_card_list
H/w O/p
ON
OFf
CO
n
LN
n
IF
ON
OF
DR
ON
OF
TR
complex condition
TD
nnnnn
NS
nnnnn
FO
AS
BI
TO
DE
DI
PR
DK
GP
S0
S1
S2
BF
RA
TO
DE
GP
S0

Note : tc__card__list : refers to the card and is selected by the first channel of that card, ie. 21,41,61 etc.

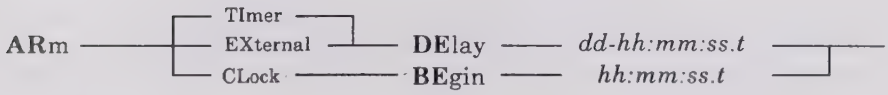
tc__chan__list : refers to the individual channel or groups of channels and is selected by the channel number, ie. 21,22-24,31-33 etc.



Function **MO** : Specify the mode of operation for the transient capture module.

IN The interactive mode allows you to select the results to be output. Once the results have been captured the SI35351F outputs an information message (if enabled) and awaits your decision. The export command can then be used to output the results and an ARm command to rearm the module ready for the next transient.

ST The standalone mode is intended mainly for unattended operation; for example in the capture of occasional transients over a long period. Measurements are made and the resulting data output automatically: the module may then be rearmed immediately, or rearmed under the control of the clock, timer, or a logging task.



Function **AR** : Arm transient capture module. (Standalone only).

TI Arm from internal timer, set delay time required.
EX Arm externally from the execute command, set delay time required.
CL Arm from internal 24 hour clock at the same time specified by the BEgin command

DE Specify delay. Default value 00-00:00:00.0
dd-hh:mm:ss.t Delay time required.

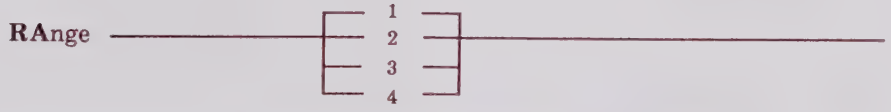
BE Specify start time. Default value 00:00:00.0
hh:mm:ss.t Start time for arming from internal clock.

Function **NU** : Specify the number of captures to be made.

* Unlimited.
n 1 to 65535

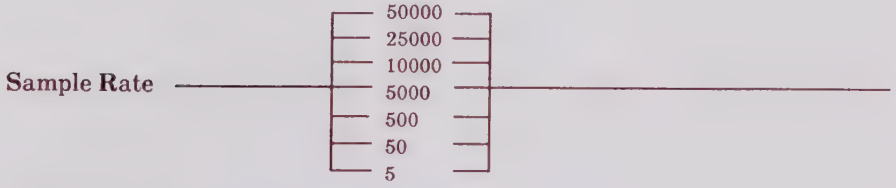
Note that if this results in large quantities of data being captured then the output device (eg. disk) may become full , (disk in normal mode).

Function **CM** : Specify the individual channel mode of operation for the transient capture module. Default is Off.

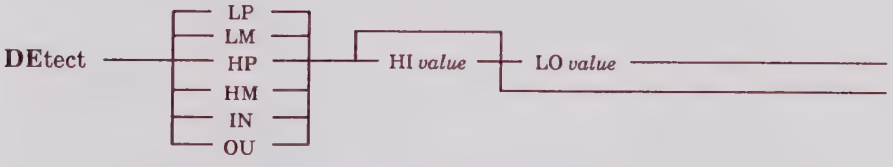


Function **RA** : Specify measurement range. Default is 10V range.

1 10V range
2 1V range
3 100mV range
4 10mV range



Function **SR** : Specify the sample rate. Default is 50,000 samples per second.

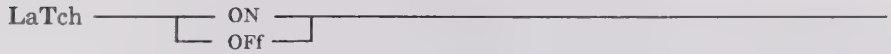


Function **DE** : Specify the condition for the detection of a trigger.

LP Positive Threshold. Specify the upper limit (HI value).
LM Negative Threshold. Specify the lower limit (LO value).
HP Positive Threshold, with Hysteresis. Specify the upper and lower limits.
HM Negative Threshold, with Hysteresis. Specify the upper and lower limits.
IN In window. Specify the upper and lower limits.
OU Out window. Specify the upper and lower limits.

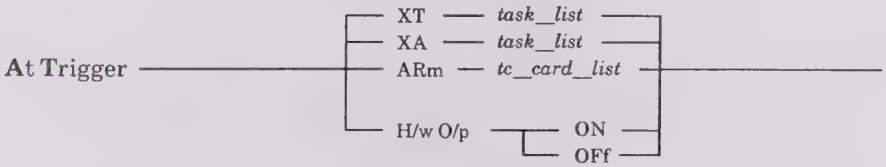
Function **DC**: Specify the number of valid samples before trigger.

n Number of samples before trigger.



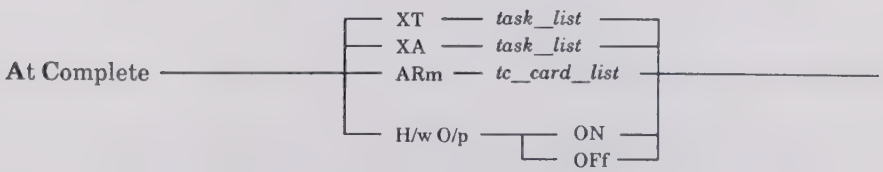
Function **LT**: Specify if latched trigger condition is required.

ON Latch condition applied on triggering.
OF No latch condition applied.



Function **AT**: Specify action on capture trigger complete.

XT Enable a task. Specify the task number. (0=off)
XA Abort a task. Specify the task number. (0=off)
AR Arm another transient capture module. Specify the module by the number of its first channel, ie, 21, 41, 61.....etc. (0=off)
HO Specify if output of hardware trigger is required.

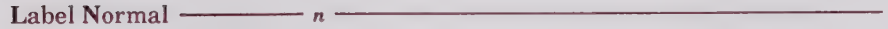


Function **AC**: Specify action on measurement complete.

XT Enable a task. Specify the task number. (0=off)
XA Abort a task. Specify the task number. (0=off)
AR Arm another transient capture module. Specify the module by the number of its first channel, ie, 21, 41, 61.....etc. (0=off)
HO Specify if output of hardware trigger is required.

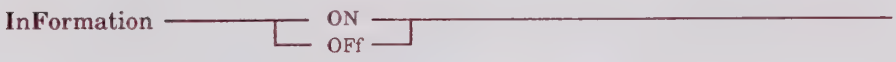
Function CO : Specify the conversion to be applied to the transient capture result.

n	1 to 600: Conversion routine number, default is 0.
0	Off



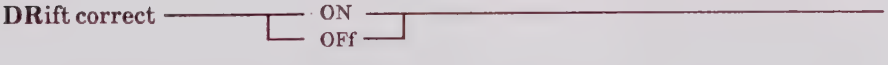
Function LN : Specify the number of the transient label.

n	1 to 600: transient label number, default is 0.
0	Off



Function IF : Specify if status messages are required.

ON	Sends one of four messages, as appropriate.
OF	No information given.



Function DR : Specify if automatic drift correct is required. Default is ON.

ON	Drift corrects on arming of transient capture module and after the channel has completed its measurement.
OF	No drift correct applied.

TRigger ——— *complex condition* ———

Function **TR** : Specify the complex condition which may contain up to three AND combinations ORed together, ie:

(D_{A1}) OR (D_{A2}) OR (D_{A3})

where D_{An} is either a single detector channel Dn or an AND combination of the form:

(Dn AND Dm)

where Dn and Dm can be D1, D2...D5 (D5 is the external TTL trigger input). Note (NOT D2) etc can also be used.

For example, detector channel 2 capture trigger could be set for,

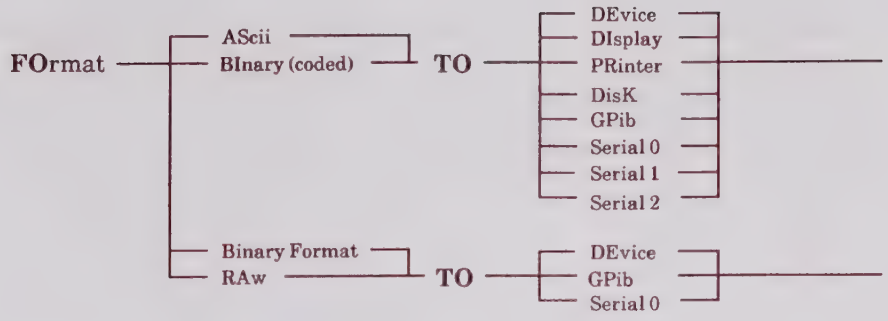
(D1 AND D2) OR (D3 AND D4) OR (NOT D5).

Trigger Delay ——— *nnnnn* ———

Function **TD** : Specify the number of input samples between occurance of the trigger and the start of data capture. If a negative number is entered, then the captured data wil begin before the trigger, ie, pre-trigger. Default is 0.

Number Samples ——— *nnnnn* ———

Function **NS** : Specify the number of results to be stored. Default is 0.



Function **FO** : Specify the format in which the data is to be output to the specified interface.

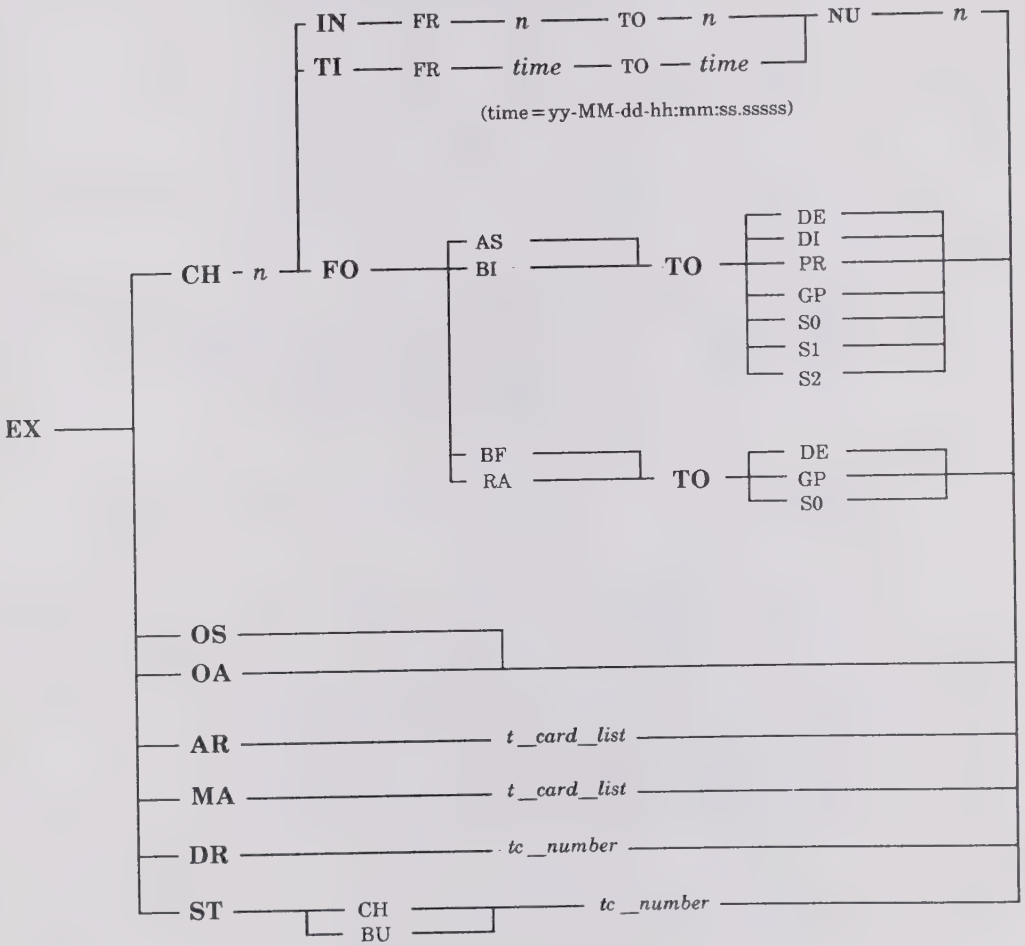
AS	Full format: 20 ASCII characters. It includes text messages in addition to full numeric information.
BI	Coded format: 10 ASCII characters. It contains results in binary format which have been encoded into printable ASCII characters.
BF	Binary format - Serial Interface and GPIB only.
RA	Raw Format - Serial Interface and GPIB only.

TO : Specify device to which data is to be logged. Default value is DE.

DE	Null device, data is lost.
DI	Front panel or local terminal.
PR	Front panel printer, (SI3535D) or printer attached to VT220 connected to SI3535F/G.
DK	Front panel disk drive SI3535D & G.
GP	GPIB interface socket on the GPIB board 35304A.
S0	The standard RS232 interface on the controller board.
S1	Socket PL1 on the serial interface board 35304B.
S2	Socket PL2 on the serial interface board 35304B.

2.23 EXPORT COMMAND

COMMAND TREE STRUCTURE :

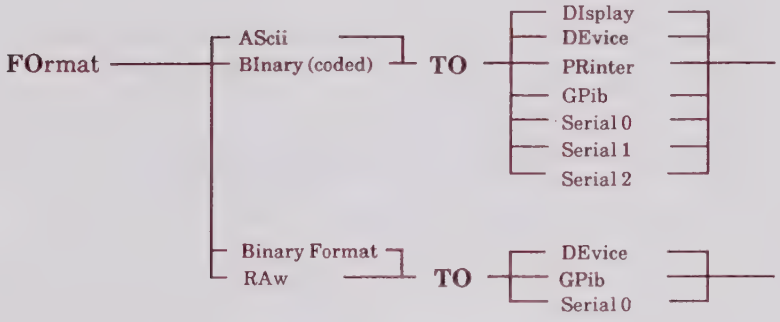


The command tree shows the commands associated with the Export Command. The Export mnemonic, EX must always be entered; however all other commands are optional and may be entered in any order (or omitted).

Function **EX** : The Export mnemonic

CHannel — *n* ————————

Function **CH**: Specify the channel number for exporting results.



Function **FO**: Specify the format in which the data is to be output to the specified interface.

AS Full format: 20 ASCII characters. It includes text messages in addition to full numeric information.
BI Coded format: 10 ASCII characters. It contains results in binary format which have been encoded into printable ASCII characters.
BF Binary format - Serial Interface and GPIB only.
RA Raw Format - Serial Interface and GPIB only.

TO: Specify device to which data is to be logged. Default value is DE.

DE Null device, data is lost.
DI Front panel or local terminal.
PR Front panel printer, (SI3535D) or printer attached to VT220 connected to SI3535F/G.

GP GPIB interface socket on the GPIB board 35304A.
S0 The standard RS232 interface on the controller board.
S1 Socket PL1 on the serial interface board 35304B.
S2 Socket PL2 on the serial interface board 35304B.

INdex —— FRom – *n* —— TO — *n* ——— NUmber — *n* ——

Function **IN** : Specify the index numbers of the first and last result required.

NU Specify the number of results to export.

TIme – FRom — *time* — TO — *time* ———— NUmber — *n* ——

(time = yy-MM-dd-hh:mm:ss.sssss)

Function **TI** : Specify the time that the first and last results required were measured.

NU Specify the number of results to export.

Output Start ————————

Function **OS** : Specify the start of result output.

Output Abort ————————

Function **OA** : Specify abort of result output.

ARm ———— *t_card_list* ————

Function **AR** : Select the arming of transient card(s). Card chosen by selecting the first channel number of required card, ie, 41, 61...etc.

Measure Abort —— *t_card_list* ————

Function **MA** : Aborts measurement on the specified transient card. This causes defined channels on the card to stop looking for triggers and to output their results immediately. Card chosen by selecting the first channel number of required card, ie, 41, 61...etc.

DRift correct now —— *tc_number* ————

Function **DR** : Select drift correct now and the channel number for drift correct.

Function **ST** : Select the status of the channel or of the buffer.

CH Channel. Returns the status of the channel, ie. armed, triggered, etc.

BU Buffer. Returns the first 6 lines of the header from the results.

Chapter 4
SI3535 Processing Language

Section		*Page*
1	**Introduction**	4.3
1.1	Processing Facilities	4.3
2	**Processing Programs**	4.4
2.1	Program Structure	4.5
2.2	Processing in Task Sequence	4.5
2.3	Statements	4.7
2.4	Syntax and Error Information	4.7
2.5	Creating a Simple Program	4.8
2.6	Producing Program Listings	4.9
3	**Program Constants and Variables**	4.10
3.1	Program Constants	4.10
3.2	Simple Variables	4.10
3.3	Indexed Variables	4.11
3.4	Pre-defined Variables	4.11
3.5	User-created Variables	4.14
3.6	Global Variables	4.14
3.7	Applications	4.15
4	**Control Structures**	4.17
4.1	IF....THEN Statements	4.17
4.2	ELSE...ENDIF	4.18
4.3	FOR...NEXT Loops	4.18
4.4	GOTO-Unconditional Branching	4.19
5	**Output Statements**	4.20
5.1	PRINT Statement	4.20
5.2	PRINT Statement Extensions	4.21
5.3	WRITE Statements	4.22
5.4	KILL Statement	4.24
5.5	SET...TO Statements	4.25
5.6	Output Format Declaration	4.28
5.7	Result Numbering	4.29
5.8	Comparison of Print Statement Extensions and Output Format Declaration	4.30
6	**Mathematical Operators and Functions**	4.31
6.1	Mathematical Operators and Priorities	4.31
6.2	Trigonometric and Exponential Functions	4.32
6.3	Relational and Boolean Operators	4.33

Section *Page*

7	**Entering and Editing Programs**	4.34
7.1	Entering Programs	4.34
7.2	Single and Multi-Statement Lines	4.34
7.3	Statement Separators/Delimiters	4.35
7.4	Expressions and Assignments	4.35
7.5	Processing Keys	4.36
7.6	Editing Keys	4.36
7.7	Resequencing	4.37
8	**Digital Inputs**	4.39
8.1	Single-Bit Inputs	4.39
8.2	Grouped Inputs	4.40
9	**Task Cross.Control**	4.41
9.1	Triggering Other Tasks	4.41
9.2	Aborting Other Tasks	4.41
10	**Counters and Timers**	4.42
10.1	Counters : COUNT1 to COUNT8	4.42
10.2	Timers : TIMER0 to TIMER8	4.43
11	**Random Channel Access**	4.44
11.1	Scanned Channel Variable - Application	4.44
12	**Storing and Replaying Processed Data and Programs**	4.46
13	**Memory Capacity**	4.47
13.1	Processing Program Memory Requirements	4.47
14	**Processing Programming Techniques**	4.48
15	**Example Programs**	4.49

Tables

4.1	SI3535 Program Constants	4.10
4.2	Mathematical Operators and Functions	4.31
4.3	Trigonometric and Exponential Functions	4.32
4.4	Relational and Boolean Operators	4.33

Illustrations

4.1	SI3535 Processing - Schematic Diagram	4.4
4.2	Task Data Flow Chart	4.6
4.3	Processing Timing Diagram	4.6
4.4	Results Output from Pre-defined Variables	4.12

1 INTRODUCTION

The SI3535 Processing Program Language provides a task-oriented data manipulation facility for processing measured data, before it is logged to an output device. Programs written in the Processing Language BASIC, can be used to perform both simple and sophisticated manipulation of channel data. Users conversant with BASIC can probably omit reading sections 4, 5 and 6, although the information may provide a useful reference.

Since processing programs are included as part of a task, for a program to be performed, it must be selected during the Task Definition (Chapter 3). Processing is only performed on channel data associated with a task defined during Channel and Task Definition, and a separate program can be appended to each of the 8 tasks.

For the SI3535D, programming can be performed using the Program Menu Key and the softkeys on the loggers front panel. Programs can be written from the front panel, where a full screen editor is available, (see user manual) or by text entry using the GPIB interface or RS232 interface. The full screen editor is available for "blind" SI3535F and SI3535G, via the front panel emulator or a VT220 terminal connected to the monitor interface.

For the SI3535D and SI3535G, using the Control Menu Key, programs can be optionally saved to disk, and subsequently re-loaded with the integral disk drive.

1.1 PROCESSING FACILITIES

Processing facilities provide the following data processing facilities these apply only to the specific channels, when used in the relevant task:

a) Modification of individual channel results, by various mathematical operations such as scaling, logarithmic conversions and trigonometric functions. These operations can be used to augment or replace conversion facilities which can be used to augment or replace the "Channel Definition" conversion facilities.

b) Cross channel calculation, such as ratios and multiplication (eg V $\times$ I) for power measurement.

c) Time calculations, such as averages, statistical behaviour, control logarithms.

d) PRINT statements for display and printing of selected results, or to send these results to peripheral interfaces.

e) Addition of explanatory text to stored results, to make them more meaningful.

f) Cross-triggering and cross-aborting of tasks under special circumstances. Control of analog and digital outputs.

g) Decision processing by use of IF...THEN statements.

2 PROCESSING PROGRAMS

The SI3535 processing programs are written in the high level language BASIC, for manipulation of channel data at various levels of complexity. Although no specific programming experience is assumed, users familiar with BASIC should recognise similarities with many SI3535 statements.

Each processing program acts exclusively on the results from the channels scanned in the Task containing that program (this is in contrast with the SI3535 Conversion facility, in which a conversion for a given channel is applied to the results from that channel in every task it scans).

For all processing, the following two main features (Figure 4.1) are applicable:

a) Each channel result is stored in a completely independent location. Hence if all eight tasks scan the same channel, eight independent results are stored, each one accessible only to the relevant Task.

b) No channel result can be processed until that channel has been declared in the relevant task.

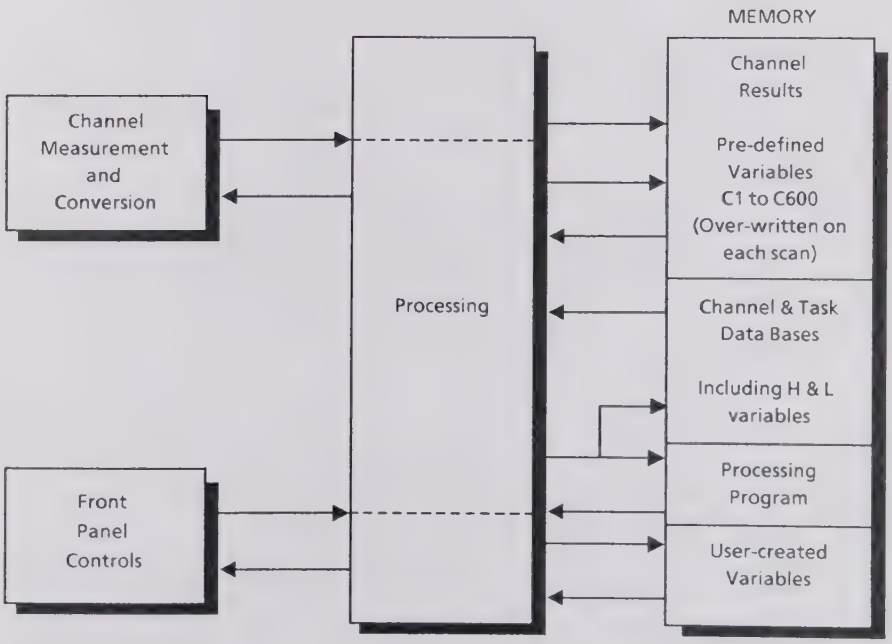


Fig 4.1 *SCORPIO Processing - Schematic Diagram*

2.1 PROGRAM STRUCTURE

Processing programs consist of consecutively numbered statement lines. During execution, instructions in the statements are obeyed in the order of their line numbers, unless the program statements indicate to the contrary. Each statement consists of control words and familiar mathematical notation, entered by processing softkeys and keypad characters.

The first item of each program line must be a 4-digit number in the form **Tnnn**:

T is the first digit indicating the number of the task, of which the program is part.

nnn are the remaining three digits, identifying the line as one of a series of program lines.

Valid line numbers are in the range 1000 to 8999, and identify the input to the SI3535 as part of a processing program, and not a Command Language input. The line number also identifies the task with which the program is associated and can be used as a label or address for movement within that program.

2.2 PROCESSING IN TASK SEQUENCE

The Scan Task Definition is used to determine whether or not measured data is to be processed. A display prompt indicating "Select end-of-scan processing" and two softkeys "On" and "Off" allow user-selection of processing. If "On" is pressed, processing will be performed on channel readings within that specific task only. The processing of channels in the scan task (Figure 4.2) is completely independent of the processing of the same channels in any other task.

It is important to note that it is only the decision whether to process data or not that is included in the Task Definition. The actual processing program can be written in the SI3535 memory at any time. Assuming processing was originally selected "Off", a program written at a later time can still be incorporated into the task by HALTING the SI3535, recalling "Select end-of-scan processing" and selecting "On". Processing can be deleted from the Task in a similar way, by setting Select end-of-scan processing to "Off". However the program is still retained in memory, and can be re-incorporated into the Task.

No procesing is performed until all channels in a specific Task have been scanned (Figure 4.3). When the SI3535 is in a Halt condition, pressing RUN again has the following effects:

a) The processor immediately reads the programs and performs:

   (i) Check of syntax error.

   (ii) Allocation in memory for user-created variables.

b) Following program checking, and any delay included in the Task Definition, scanning starts.

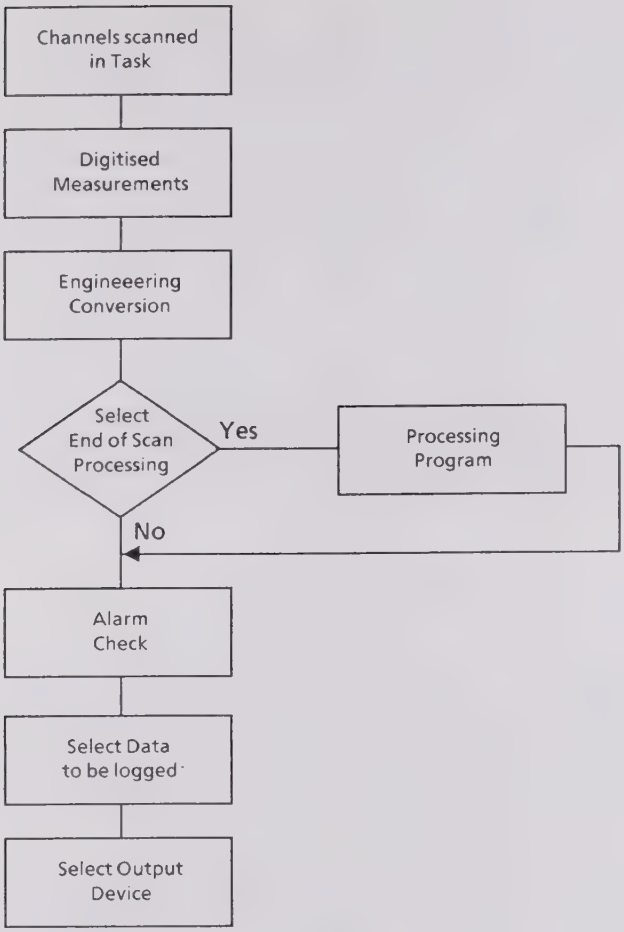


Fig 4.2 *Simplified Data Flow Chart*

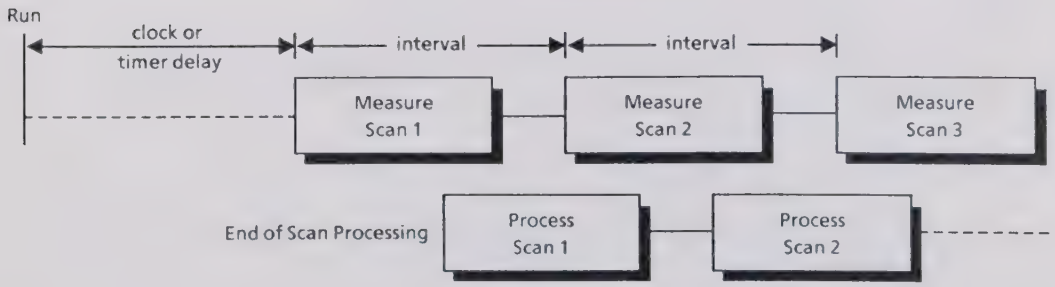


Fig 4.3 *Processing Timing Diagram*

2.3 STATEMENTS

In general, statements consist of one or more reserved control words (section 4), and an expression; each statement uses English words and familiar mathematical notations.

eg **1050 IF CI < > 10 THEN:GOTO1100:ENDIF**

This line contains the following 3 statements:

Statement 1 : **IF CI < > 10 THEN**
Statement 2 : **GOTO 1100**
Statement 3 : **ENDIF**

For multi-statement lines, each statement must be separated by a colon.

Note 1: Colons are not required following either an IF....THEN statement or carriage return (C/R), but will be ignored if used.

2.4 SYNTAX AND ERROR INFORMATION

Syntax provides the processing language rules, defining the form of programming statements. It is essential that the syntax is adhered to, for correct interpretation of programs by the processor, (eg. semi-colons, spaces etc may be essential parts of a statement and not merely punctuation). The general form for the main control structures is given in section 4. General syntax rules are defined as follows:

a) **SPACES** : These **must** be included in program lines only where specified:

(i) **PROCESSING KEYS** : The sixteen words provided by the PROCESSING softkeys are automatically preceded and followed by a space. If these words are entered letter by letter from the keypad, which is perfectly permissible, two spaces must be included.

(ii) **LINE NUMBERS** : Program line numbers are automatically followed by a space when they appear in the display after pressing ENTER but when in remote entry a space needs to be added. Hence a space should not be added solely for the purpose of separating the line number from the rest of the line.

(iii) **PRINT STATEMENT** : The word PRINT is always followed by a space. When using a modifier with a Print Statement, the line, as typed, will appear in the Display as eg PRINT >. However, if the line is subsequently recalled to the display, the processor will have "stripped" the space out, so that PRINT> appears.

(iv) **EXPRESSIONS** : Expressions need not include spaces around the operator, eg type C1 > 4 not C1 > 4.

(v) **VARIABLES** : Refer to b), (ii).

(vi) **INVERTED COMMAS** : Spaces are perfectly acceptable anywhere within inverted commas.

(vii) **MINUS SIGNS** : Except where the minus sign is used to indicate subtraction, or a negative exponent, it must be preceded by a space,
eg a negative constant must be typed (space)-1.29E-6.

b) **VARIABLES**

(i) Within the limits on length specified in section 3 the names of variables can consist of any mixture of alpha numeric characters except that they must never begin with a number, eg **TEMP4** is acceptable, **4TEMP** is not.

(ii) The names of variables must not included "embedded" spaces, since the processor cannot recognize them. For example, **C21** is acceptable, **C 21** will be rejected and produce Error Code 159 **Illegal Variable**.

c) **PRINT STATEMENTS**

(i) Print Statements containing an operator and an indexed variable must never start with the variable,
eg **PRINT PRESSURE(I)*0.7** is wrong, **PRINT 0.7*PRESSURE(I)** is correct.

(ii) Refer also to rule a), (iv) regarding spaces.

d) **MULTI-STATEMENT LINES**

(i) Multi-statement lines must include a colon as a separator between each pair of statements, as described in section 5.

2.4.1 Error Checks

To prevent invalid statements corrupting the final program, three stages of checking are automatically performed, as follows:

a) As each new line of program is entered, it is checked for syntax errors within the line.

b) When **RUN** is pressed, the processor scans through the program before running it, to assign storage space for the variables etc. During this process, syntax errors which may be spread over several lines, eg in **FOR....NEXT** loops, are detected.

c) When the program is actually running, more complex errors are detected, (eg that a Channel to be processed is not present in the associated Task).

Errors of the types described above are listed in Appendix 3, "Error Messages". Items a) and b) are classified as "Processing" error, item c) are "Run Time" errors.

2.5 CREATING AND RUNNING A SIMPLE PROGRAM

To gain familiarity with the processing program controls, enter a simple program as explained in detail below. The object of the program is to take the result, which has been stored during the scan in the pre-defined variable **C1**, multiply it by 10, then store the new, processed result in the user-created variable **A**. The processed result is then printed. The lines must be typed exactly as shown, for example, do not press the SPACE key unless so instructed.

```
1000 A=C1*10
1001 PRINT A
```

This program assumes that Task 1 scans a d.c. voltage on Channel 1. Any keying errors can be rectified by using the EDIT keys. Proceed as follows:

1. Press the PROGRAM key.

2. Select the ETC. softkey, then press the PROCESS softkey.

3. The bottom line of the prompt Display will now show:

   "Enter line to start editing"

Type the line number 1000 on the keypad, whereupon 1000 will appear in the bottom line of the Display.

4. Press ENTER. The line number 1000 will move to the top line, leaving the rest of the Display clear to receive the program information.

5. Type **A = C1*10** on the keypad, remembering to use the SHIFT key before and after **A** and **C**. The characters will appear in the Display as they are typed, following the line number 1000.

In this line of program, a new variable called **A** has been created by the user and the processor will automatically allocate a place in the memory for it. The line means: "The value stored in A becomes the value of Channel 1 multiplied by 10".

6. Presss ENTER. The previously displayed line will be transferred into the processor memory and the next line number, ie 1001, will automatically appear on the next line.

7. Press the PROCESSING key PRINT followed by the keypad key A. All the PROCESSING key words are automatically followed by a space; there is no need to add one. Press ENTER.

2.6 PRODUCING PROGRAM LISTINGS

To obtain a printed record of the program, proceed as follows:

1. Press the Control Menu key.

2. Press the List softkey.

3. Press the softkey Program.

4. The display will show: **Enter program line numbers**

5. Type **1000-1001 ENTER**

6. The display will show: **Select output device**

7. Press the softkey Print

8. The printer will produce:

   1000 A = C1*10
   1001 PRINT A

Note 1: If, in Step 5, the ENTER key is pressed without first declaring any line numbers, all programs in the memory will be listed.

3 PROGRAM CONSTANTS AND VARIABLES

3.1 PROGRAM CONSTANTS

A program constant is defined as a data item which does not change. The types of constant recognized by the processing program are summarized in Table 4.1.

Table 4.1 *SI3535 Program Constants*

TYPE	EXAMPLE	DESCRIPTION
Numeric	123.4567	Numeric constants, upto 7 significant figures.
	0.13	Decimal constants must have leading zeroes included.
Exponential	-1.234E-15	Exponential format constant, upto 4 significant figures in a 10 character field. This format is automatically selected for an output by the SI3535 if either : (a) 4 or less significant figures would be displayed or (b) The value is greater than 1E8. Note: Except where the minus sign is used to indicate subtraction, or a negative exponent, it must be preceded by a space.
Text	"RESULT"	Text to accompany a processed result in Full Format; can be upto 18 characters.
	"PSI"	Units to accompany result; can be up to 3 characters.
	"OVERLOAD"	A message can be up to 38 characters in Full Format and up to 18 characters in Compact Format.

3.2 SIMPLE VARIABLES

A variable is a name associated with a specific memory location for storage of a data item; the value of the data item can change. Simple variables comprise one element, and can store only one piece of information (eg. the result from a single resistance thermometer could be processed, and stored in a variable called **TEMP**).

In a program using simple variables, each piece of information to be stored must have a separately allocated variable. Hence, simple variables are mainly used when processing relatively few results, or where a small number of different transducers are used with a proportionately large number of otherwise identical ones.

Simple variables are not declared at the start of the program, and can be introduced at any point.

3.3 INDEXED VARIABLES

Indexed variables consist of a number of elements which make up an array. Each element in the array has a unique address, comprising the variable name together with element number. The generalized form of an indexed variable is as follows:

VARIABLE (I)

in which **VARIABLE** is a name, and **I** is an index number which must be defined in the program.

Indexed variables allow access to large numbers of results in the most direct way, by referring to all elements in an array in one brief statement. The program could become quite lengthy if implemented using simple variables.

Each element in an array can be addressed individually (eg A(1), A(2) etc). Brackets are mandatory, and the processor recognises that A(1) and A1 are totally different.

3.3.1 Declaring Indexed Variables

The name and number of elements in each indexed variable must be declared at the start of the program. Each declaration is effectively a statement, and allows the processor to reserve the required number of memory locations. A typical declaration in line Tnnn is given below:

Tnnn MEAN;5

The processor is informed that there is an indexed variable called **MEAN**, processing 5 separate elements called MEAN(1), MEAN(2) etc (note the semi-colon separator).

3.4 PRE-DEFINED VARIABLES

Pre-defined variables have names permanently reserved for them in the SI3535 memory. These variables have been specially developed, and share certain properties of both the simple and the indexed forms. Strictly speaking, they constitute three indexed variables, named C, H and L. Each of these variables can contain up to 600 elements to conform to the maximum 600-Channel capacity of the SI3535.

By the rules of indexed variables, each element would normally be referred to in the program as, eg C(1), H(50), L(300) etc. However, since these variables are used very frequently (particularly the C1 to C600 group), the processor has been permanently programmed to recognise them in the simple variable form (eg C1, H50, L300). Unlike other indexed variables, the pre-defined variables do not have to be declared at the start of the program. However, the normal indexed form can be used in programs (eg in a **FOR...NEXt** loop). For example, the following section of program will cause the results of the first 200 Channels to be output. H(I) and L(I) are also recognised by the processor, as the High and Low limits respectively of Channel (I).

```
7054 FORI = 1 TO 200
7055 PRINT C(I)
7056 NEXT
```

The features are common to all three groups of pre-defined variables. However, the variables C1 to C600 also differ in certain respects from the variables H1 to H600 and L1 to L600, as detailed in the two following sub-sections.

3.4.1 Variables C1 to C600

a) The programmer has no control over the storing of the original results in these variables, since this is performed by the processor as the Channels are scanned. Each result is automatically placed in the correct location, eg. Channel 6 result is always stored in the pre-defined variable C6. However, the programmer does have a choice when processing these results, as to whether the intermediate, or fully-processed, versions shall be stored back in the pre-defined variables, over-writing the original result, or in user-created variables.

b) The Channel results cannot be processed as part of a task unless they have already been declared in that task.

c) These variables are not global. Each one is accessible only to the program contained in the task which scanned the Channel.

d) The processor recognises the names C1 to C600 in all the 8 tasks, but this does not mean that there are 8 × 600 locations permanently available. There is a shared pool of buffers which act as variables for storing the Channel results. As each result is obtained, a buffer in the pool is assigned the relevant name (eg. C1), and the result is stored in it. The buffer retains the identity C1 only until processing is complete, whereupon it becomes available for re-use. For the majority of applications, results are processed fast enough to keep a reserve of unassigned buffers in the pool. However, in exceptional circumstances, (eg. a fast scanning rate coupled with extensive processing) all the buffers may have been filled. In this case, the scanning rate of the logger automatically slows down until buffers become available.

e) Any results stored in these variables, whether processed or not, are over-written on the next scan (ie. the original values are replaced by the new ones).

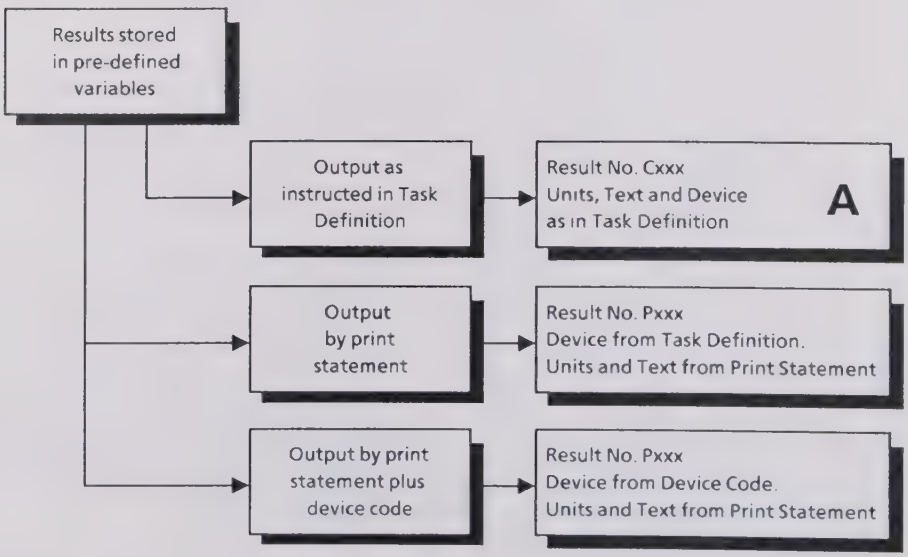

Results stored in pre-defined variables
Output as instructed in Task Definition
Result No. Cxxx
Units, Text and Device as in Task Definition
A
Output by print statement
Result No. Pxxx
Device from Task Definition.
Units and Text from Print Statement
Output by print statement plus device code
Result No. Pxxx
Device from Device Code.
Units and Text from Print Statement

Fig 4.4 *Results Output from Pre-defined Variables*

Note: When the Print Statement is used, with no additional processing such as PRINT C1*10, it merely copies the result that is printed in "A". If result "A" is not required, the KILL statement, may be used.

3.4.2 Variables H1 to H600 and L1 to L600

a) The values stored in these variables are used as High and Low alarm limits on the Channel results, eg. H1 and L1 are the High and Low limits for Channel 1, H2 and L2 for Channel 2 etc. The values are usually entered initially during Channel Definition.

b) The values entered in the variables are read by the processor as soon as RUN is pressed but before scanning commences. When the logger is halted, the values can be changed by entering new ones in the Channel Definition.

c) Unlike C1 to C600, the contents of these variables are not over-written on each scan.

d) The values can also be changed by means of a program statement. For example, the High Limit on Channel 12 may have been set to 7.9 by the Channel Definition. The program could then include a statement to warn the user when Channel 12's result exceeds 7.9, but as this may not be dangerous, the limit could then be automatically increased to 8.9 ready for the next scan.

e) These variables are global.

f) Limits which are unique to a particular task can effectively be created by using the **IF...THEN** statement mentioned in item 4 above. For example, task 3's program could contain the line:

3330 IF C16>3.14 THEN

followed by further lines detailing the appropriate action, eg. show a warning message in the Display, or Print an alarm message.

This type of alarm limit does not affect the results from Channel 16 in any of the other tasks because it has not changed H16.

g) These variables are the only ones to be stored in a special area of memory, called the database and it is because of this that their contents exist only as numbers, with no text, units etc.

h) Care must be taken if they are used for storing anything other than High or Low limits, since, unlike other variables, they are located in the database. Therefore, the amount of space allocated to them varies according to the type of Input/Output modules fitted.

j) A logger which is configured for n Channels, automatically has the requisite n H and L locations reserved in the database. If the configuration is subsequently changed to n + m or n - m Channels, the H and L locations are automatically altered accordingly.

3.5 USER-CREATED VARIABLES

All the variables that the SI3535 can recognise are user-created, with the sole exception of the pre-defined ones. Hence, whether the variable is simple or indexed, it can be created only by the user giving it a name in the program.

As long as a program remains in memory, any user-created variables named in it will be re-created by the processor each time that RUN is initiated to start scanning. This process occurs for every program stored in the memory, even if "Select end-of-scan processing" is "Off" for any or all of the tasks. This has the advantage that any indexed global variables, that are declared in a program which is swiched "Off" will still be created in memory by the processor. Hence, these variables are still available for use by the programs in other tasks, without having to be declared again.

3.6 GLOBAL VARIABLES

A global variable is accessible from any task rather than just the one in whose program it first appears. All the variables that the SI3535 can recognise are global, with the exception of the pre-defined variables C1 to C600; each Channel result, such as C1, is only accessible within the task that scans it.

Note 1: H1 to H600 and L1 to L600, although also pre-defined, are nevertheless global.

The user must take care when using global variables that a statement in one task does not inadvertently affect the performance of other tasks. For example, the simple variable MEAN, which is global, may have been used in task 1 to store the value of the expression:

(C1 + C2 + C3)/3

The programmer may re-use MEAN again in task 8, to store the value (C8 + C9 + C10)/3. The processor is now asked to store two totally different results in the same variable! It solves this problem by over-writing whatever is in MEAN with the result derived from the latest task performed.

Although this may be satisfactory for that task, because of the multi-tasking nature of the SI3535, it may yield a totally invalid result for the other one. Also, if such an inadvertently-shared variable is used for totalizing over several scans, it will give a misleading result in both tasks. The processor cannot identify the presence of a shared variable as a syntax error, as it may have been introduced deliberately as a perfectly valid programming technique. To guard against this, the task number could be used as a suffix for all variables contained in it, eg MEAN1 for task 1, MEAN5 for task 5, etc.

I is used throughout this Chapter to represent the index in the general form of various statements, such as indexed variables. For example, TEMP(I), where the number of elements represented by I must be declared in the program as eg, TEMP;5. This variable could be included in a FOR...NEXT loop, where I can be included in the program.

Note 2: I is a global variable, so can be used in only for FOR...NEXT loop in one task. This problem can be overcome by using suffices, eg, I1, I2 etc, or even A, B, D etc, instead of I.

3.7 APPLICATIONS

The following list of features is included to assist in deciding which of these two categories of variable is best suited to a particular applications. The comments on pre-defined variables apply whether the results stored therein have been processed or not.

1. Making Channel results global

   A result contained in one of the pre-defined variables C1 to C600 is accessible only to a program contained in the task that originally scanned it. However, if the result is transferred from the pre-defined variable to a user-created variable, with or without any other processing, eg. by writing A=C1, this new variable A, is global, and therefore accessible to all tasks.

2. Totalizing over several scans

   Results stored in pre-defined variables are over-written on each scan. Therefore these variables cannot be used for totalizing over several scans. User-created variables, although cleared when RUN is initiated to start scanning; can be used to accumulate results over all the scans that occur until the RUN/HALT key is pressed again to halt the logger.

3. Selecting the output device

   Pre-defined Variables

   a) Results are always sent to the output device specified in the Task Definition.

   b) Results can be copied to a different device by means of a Print Statement which includes the code for that device.

   User-created Variables

   Results stored in user-created variables cannot be sent to an output device at all until they have been included in a Print Statement, which may take one or the following two forms:

   a) A Print Statement with no device code included; this will simply send the results to the output device specified in the Task Definition.

   b) A Print Statement which included a device code; this will send the results to the device represented by that code.

4. Information accompanying the output result

   Pre-defined Variables

   Results stored in pre-defined variables can be accompanied by the following information:

   a) As previously stated, results are always sent to the output device specified in the Task Definition, and are accompanied by any text and units that are included in the Channel Definition.

b) If a result is also included in a Print Statement, with no device code and no extensions, it will be copied to the device specified in the Task Definition. Therefore two identical results will appear as follows:

   (i) The result from the pre-defined variable, as in a) above.

   (ii) A duplicate result produced by the Print Statement, but with no accompanying text. Hence, there is seldom any point in using this method.

c) If a result is included in a Print Statement with a device code but no Extension, it will be copied to the specified device with no additional information.

d) If a result is included in a Print Statement, with a device code and Extensions, it will be copied to the specified device accompanied by the information included in the Print Statement Extensions.

User-created Variables

Results stored in user-created variables fall into two categories, both of which require a Print Statement in order to appear as output results at all:

a) Simple variables. If text, unique processed results number, and units information are required to accompany the results, they must be included in Print Statement Extensions.

b) Indexed variables. Information, as described for simple variables, can also accompany results stored in indexed variables. However, there are now two possible sources of this information (ie. Print Statement Extensions and Output Format Declarations). Every program which introduces an indexed variable must declare it at the top of the program. This declaration can be extended to include results to an output device, the program must contain a Print Statement, which can similarly be extended to include text, units etc.

4 CONTROL STRUCTURES

Control structures effectively use reserved words within the SI3535 processing program language to produce statements on program lines.

4.1 IF....THEN STATEMENTS

Although a simple program may progress from line to line, performing each instruction in sequence, practical processing usually requires decision making and consequent action(s). By means of an **IF....THEN** statement, you can instruct the processor to choose, conditionally, between two different program paths. The point in the program at which the decision is taken is called a "branch", the simplest structural form being as follows:

IF relational expression **THEN** statement : **ENDIF**

For example,

```
4008 IF C1>5 THEN

4009 TRIGGER 6

4010 ENDIF

4011...etc
```

In the example, line 4008 contains the expression C1>5, comparing the value of Channel 1 with constant 5.

This comparison provides one of two results as follows:

a) TRUE - indicating that the Channel 1 value is greater than 5, and actioning **TRIGGER 6**, which causes task 6 to run concurrently with the original task. The processor then proceeds automatically to line 4010 where instruction **ENDIF** indicates that **IF....THEN** is complete and line 4011 can be actioned.

b) FALSE - indicating the Channel 1 value is less than or equal to 5, causing the processor to ignore line 4009 and proceed to **ENDIF**; the processor then proceeds to action line 4011.

4.1.1 Multiple Statements

The example given in Section 4.1 contains only one statement between **THEN** and **ENDIF**. However, multiple statements are also possible which can be included on the same line, or on separate lines as shown below. Reference should be made to Section 7.2 for details of restrictions applying to multiple statements.

```
4008 IF C1>5 THEN

4009 TRIGGER 6

4010 ABORT 5

4011 PRINT "TEMP>LIMIT"

4012 ENDIF
```

A TRUE result causes the 3 statements in lines 4009, 4010 and 4011 to be actioned followed. A FALSE result causes the processor to ignore all 3 statements and move directly to line 4012 to execute the **ENDIF** statement.

4.2 ELSE....ENDIF STATEMENT

Using **ELSE** together with an **IF....THEN** structure extends the branched statement options. If an expression is FALSE, the program progresses to an **ELSE** statement or **ENDIF**, and whichever is found first will be actioned. The simplest form of the structure is as follows:

IF expression **THEN** statement 1 : **ELSE** statement 2 : **ENDIF**

The example shown below, extends the previous single statement (section 4.1)

```
4008 IF C1>5 THEN

4009 TRIGGER 6

4010 ELSE

4011 PRINT "FREEZER O.K."

4012 ENDIF
```

The next line is executed on the basis of the two results as follows:

a) TRUE - causes the processor to action line 4009 as before, but also nullifies the **ELSE** option; hence the the final **ENDIF** statement is executed.

b) FALSE - causes the **ELSE** option to be actioned. Line 4011 is then executed followed by the **ENDIF** statement.

4.3 FOR....NEXT LOOPS

A program loop is a sequence of instructions that is executed repeatedly, until a specific condition is satisfied. The beginning and end of a loop are defined in the SI3535 processing instructions by **FOR....NEXT** statements. You can specify the number of times that the loop is to be repeated by using the **TO** and **STEP** commands. The simplified general form is as follows:

FOR I = x **TO** y: statement: **NEXT**

The loop variable I is initially set to value x, and the instructions are carried out. **NEXT** causes I to be set to (x + 1) and the same instructions are repeated, until I has been set to value y, at which point the instructions are performed for the last time. The example below, shows a program for printing the results of a channel set of 50 channels:

```
3005 FOR I = 1 TO 50

3006 PRINTS SC(I)

3007 NEXT
```

The loop will be repeated until I has been set to 50 when instruction 3006 will be carried out for the last time.

Nesting

FOR....NEXT loops can be nested up to 4 levels. It should be stressed that **FOR....NEXT** statements must always be paired; hence each **FOR** statement must always be terminated by its own **NEXT** statement. An example of 2 level nesting is shown below:

```
1008 FOR I = 1 TO 4      Level 1

1009 FOR J = 1 TO 3      Level 2

1010 statement

1011 NEXT

1012 NEXT
```

4.4 GOTO - UNCONDITIONAL BRANCHING

A **GOTO** statement unconditionally transfers program execution control to the line number indicated in that statement, allowing the program to progress from that line. Hence the general expression:

GOTO Tnnn means - continue execution of the program at line Tnnn

T= Number of task containing program
nnn= Line number in same task program.

NOTE: It is not possible to transfer control to a line number of a different task.

Program branching using **GOTO** can be in a "forward" or "backwards" direction through the program, but **GOTO** must be the last statement on a multi-statement line; all lines skipped-over are ignored. For transfer backwards, you should take care not to create an endless loop.

GOTO must not be used to exit from **IF....THEN** and **FOR....NEXT** structures. Line numbers involved in **GOTO** operations are automatically adjusted during resequencing (Section 7.7), following use of the **LINE INSERT** key.

5 OUTPUT STATEMENTS

The processing program must contain an output statement in order for results or processed data (other than that held in channel variables) to be output to any external device or interface. The main types of output statement are as follows:

a) **PRINT** statement allows information to be added to the task data output queue. It is generally used to output results or processed data to any selected output device.

b) **WRITE** statement enables data to be output to the selected device as soon as it occurs in the program. It is generally used for sending text messages.

c) **KILL** statement which is used to delete redundant Channel results from the output data.

d) **SET....TO** statement which is used to control analog and digital outputs from the SI3535.

A brief description of each type of statement is given in the following paragraphs.

5.1 PRINT STATEMENT

The action of the **PRINT** statement is to add a processed result to the current scanned Channel results. The processed result appears within the confines of the **Start** and **Done** task markers. The general form of the complete **PRINT** statement with extensions is as follows:

PRINT [modifier] [device] item; [extensions], [item; [extensions]]

Note: The square brackets [ ] denote optional parameters, and are not included in the actual program.

5.1.1 PRINT Item

The simplest form of the **PRINT** statement is one without extension or other optional parameters:

PRINT item

which usually takes one of the following forms:

a) **PRINT** expression. This form of the PRINT statement causes the results stored in a simple variable to be sent to the output device specified in the Task Definition. The result has no text or unit, and is labelled as processed result number 1 (ie. P001).

eg. **1005 PRINT TEMP**

b) **PRINT** indexed variable. The statement causes a result to be appended as for a), using a previously declared indexed variable.

eg. **2010 PRINT CHAN(10)**

c) **PRINT "String".** "String" represents a sequence of alpha-numeric characters, comprising a text message, which must be enclosed in quotation marks. The statement causes a "text only" result to be inserted before any existing results in the scan, but after any other "text only" results. Up to 18 characters can be used for compact format, and up to 38 for full format output.

[modifier]

The modifier is used to force the output into alarm format, when a processed result is outside the permitted limits. The results usually form part of a conditional structure.

[device code]

The device code consists of a number of codes, which direct the processed results to a particular output device. It is only necessary to include the code in a statement, if the output device required for the specific result is different to that specified for the associated task.

Action	Device Code
Print to Display	\DI
Print to RS232	\S0
Print to Printer	\PR
Print to Disk	\DK
Print to GPIB	\GP
Print to RS422/1	\S1
Print to RS422/2	\S2

5.2 PRINT STATEMENT EXTENSIONS

The basic form of PRINT statement is one without the use of print extensions, ie:

PRINT [modifier] [device] item

However, with the exception of PRINT "string", the PRINT statement can be extended to make selected items of output information more meaningful, and to enable alarm text to accompany processed results:

PRINT [modifier] [device] item [;extension] [;extension] [;extension]

Note: Each extension must be separated by a semicolon (;).

The **PRINT** statement may be extended to contain the following items of information in the specific combinations:

a) Units
b) Processed result number
c) Processed result number and units
d) Text and processed result number
e) Text, processed result number and units

Note: When using a **PRINT** extension the output will remain in the standard SI3535 output format.

PRINT extensions are particularly useful with simple variables, (ie. to add explanatory or alarm text to output). However, caution must be used when using **PRINT** extensions with indexed variables which have been defined using the Output Format Declaration. When a **PRINT** extension is used the Output Format Declaration will be completely overriden, and if not replaced by an extension item, the field will remain blank.

5.2.1 Change Processed Result Number

All processed results are numbered P001 unless otherwise defined. The print extension can be used to change the processed result number to one more appropriate (the semicolon must be used followed by the required result number).

eg. **1011 PRINT CHAN(10);10**

5.2.2 Adding Units

Units of up to three characters can be added to the processed results. The semicolon separator must be used and the units must be enclosed in inverted commas.

eg. **1011 PRINT CHAN(10);10;"VDC"**

5.2.3 Adding Descriptive Text

Text of up to 18 characters can be added to the output in Full Format. Text must be enclosed in inverted commas, and preceded by the semicolon separator.

eg. **1011 PRINT CHAN(10);"NORMAL";10;"VDC"**

5.3 WRITE STATEMENTS

WRITE statements enable text messages to be sent to the output device specified in the task Definition. A device code can be included to send the message to a different output device. The main differences between the PRINT and WRITE Statements are as follows:

a) WRITE statements are treated as text only, although this text may include a calculation, and are not in the SI3535 result format.

b) WRITE statements do not appear within the relevant Start and Done task markers. They are output as soon as possible after the processor has encountered them in the program. Their precise position in the sequence of output data is difficult to predict, owing to the multi-tasking nature of the system software, but it is likely that the message will appear before the standard scan output.

The general form of the WRITE Statement is detailed below:

WRITE[device code] item,item,item,...etc

[device code]

The device code defines the output device to which the message is to be sent, as follows:

Action	Device Code
Write to Display	\DI
Write to Printer	\PR
Write to Disk	\DK
Write to GPIB	\GP
Write to RS232	\S0
Write to RS422/a	\S1
Write to RS422/b	\S2

The code must be included in the statement only if the output device required for the message is different to that specified for the associate task. For example, if the Task Definition directs all results to the RS232 port, and the WRITE statement message includes the code for GPIB, then the message will be sent to the GPIB only.

Only one code may be used in any one WRITE statement. However, messages can be duplicated to several output devices, by repeating the WRITE statement for the message on as many lines of program as necessary, using a different device code each time. However, these messages cannot be sent to the disk of SI3535D or SI3535G in Coded format and subsequently decoded, due to their text only content.

Items

The item to be written may be one of two types:

a) An **expression**, which may be any variable or a mathematical or Boolean expression which evaluates to a numeric result.

eg For the program line below, where **C1** contains the value **5**.

3030 WRITE C1*10

This causes the value 50 to appear in the output, with no text, units, or result number; this form of Write Statement therefore has fairly limited applications. The processor knows that it is to Write the value contained in **C1** multiplied by ten, rather than treat the statement **C1*10** as text.

b) A "string" which may be any string of characters enclosed in quotation marks.

A practical application of the **WRITE** Statement, using both the above types is to include it in an **IF...THEN** statement.

```
2020 IF C21>5 THEN

2021 WRITE \DI "WARNING AT", TIMER0, "SECONDS"

2022 ENDIF
```

Assuming that all other information from task 2 is being sent to the RS232 port, line 2021 causes the numeric value accumulated in Timer 0 to be added to the text and the whole message to be sent to the display, where it will appear in the form:

WARNING AT 26.0000 SECONDS

WARNING AT 28.0000 SECONDS

etc., updated according to the scanning rate of task 2.

When the **WRITE** Statement includes several items (ie general form), a comma must be included between each pair as a separator. This comma also causes each item to be added to the output message at specific "tab" positions in the standard output format.

When printed, each number, is limited to maximum numeric field size of ten characters, plus an automatically inserted space. Each value of a succession of values in the same message will begin on a tab mark, eg:

368.....4840.....50.....etc

A **string** occupies its precise length, eg the message **RUN AWAY** occupies eight character positions.

5.4 KILL STATEMENT

The **KILL** statement can be used to delete Channel results from the data stream which is fed to the output device. This facility is useful where "**All**" data has been selected for logging in the task, but certain results are not required to be output. For example, only the **processed** results of certain Channels may be required, and hence the original readings may be discarded.

The storage space of a **KILL'd** result is made available for a fresh result **immediately** the statement is executed. This helps to maintain the scanning rate when the measurement buffer is approaching an overload condition. The general form of the statement is as follows:

KILL p,q,r,s,...etc

The number of Channels included in each **KILL** statement is limited only by the line length capacity of the display. The statement is applicable solely to the pre-defined variables **C1 to C600**. Only the element number must be quoted, eg to remove the results of Channels 1, 2 and 3 write:

KILL 1,2,3 not KILL C1,C2,C3

Where consecutive Channels are to be **KILL'd** (eg Channels 7,8,9,10 and 11), an indexed form can be used in a **FOR...NEXT** loop:

```
6066 FOR I=7 TO 11

6067 KILL (I)

6068 NEXT
```

Where the Channels are in a non-consecutive sequence (eg 5,13,19,48,92) with no other intervening numbers, the whole sequence can be **KILL'd** by using the Scanned Channel form SN(I). Using an example, if the SI3535 measures current on Channel 1 and voltage on Channel 2, the product of these (ie the power) is expressed as follows:

```
C1=C1*C2
```

This re-uses the predefined variable **C1** to store the power value, at the same time, deleting the redundant current value. This still leaves the unwanted voltage value in C2, but by now writing the following, the storage space for the original Channel 2 result is immediately made available for re-use.

```
KILL 2
```

5.5 SET...TO STATEMENTS

The SET...TO statement can be used to condition the analog and digital outputs of the SI3535 as follows:

5.5.1 Analog Outputs

The **SET analog channel TO value** instruction sets the output of specified Analog Channel to the value defined. The SET...TO instruction can either set the output to a fixed value, or cause it to follow the values of one of the program variables, as shown in the following examples. These examples assume that Channels 61 to 65 are Analog Outputs, set up as current sources:

eg.1 **Fixed Value Output:** The instruction below causes Channel 63 to produce 50% of its output.

```
1008 SET 63 to 50
```

As a current source, the output can be varied between 4 and 20 mA (ie a change of 16mA). 50% of this is 8 mA, added to the minimum of 4 mA equals 12 mA.

eg.2 **Output Controlled by Program Variable:**

```
1008 SET 63 TO SETPOINT(3)
```

Every time the program is executed, the output of Channel 63 will be set to the value currently stored in the variable **SETPOINT(3)**. This value must also be a percentage, which must have been calculated earlier in the program prior to being stored in **SETPOINT(3)**

This facility can be extended by using a **FOR...NEXT** loop:

```
1008 FOR I=1 TO 5

1009 SET (I+60) TO SETPOINT(I)

1010 NEXT
```

Channels 61 to 65 inclusive will have their outputs set to the values stored in the variables **SETPOINT(1)** to **SETPOINT(5)** respectively.

5.5.2 Digital Outputs

The state of any output of the Digital Input-Output Module can be changed by a program statement of the form:

SET digital channel TO value

: digital channel	represents the number of the required Digital Output Channel.
: value	represents an expression which the processor resolves into either Logic 1 or Logic 0.

(More than one Channel can be **SET** as shown under **SET...TO** string)

Assuming that Channels 41 to 60 are Digital I/O type. An example of a simple digital output statement is:

```
3004 SET 41 to 1
```

This statement sets the output of Channel 41 to Logic 1, (eg. a switch is closed). Conversely:

```
3022 SET 41 to 0
```

sets the output of Channel 41 to Logic 0 (eg. a switch is open).

Note: Any outputs which have been SET TO Logic 1 automatically revert to Logic 0 (ie the switches open, when the logger is halted).

The only numeric value which is unique is zero, represented by Logic 0. Every value other than zero is interpreted by the processor as being Logic 1, eg:

```
3004 SET 41 to 6-1  or:

3004 SET 41 to 36
```

are both interpreted as:

```
SET 41 to 1
```

As another example of the use of processing applied to Digital Outputs, a Channel could be used to indicate equality between two analog inputs.

5.5.3 Strings

Several consecutive Digital Channels can have their values included in a "string" instruction using **SET** digital channel **TO** "string". The line below causes an output pattern to be set up as shown:

3004 SET 41 to "10110"

Channel	41	42	43	44	45
Value	1	0	1	1	0

Notes:
1. Only the first Channel in the sequence must be specified in the instruction.
2. The inverted commas around the string are essential, **SET 41** to 10110 is interpreted by the logger as **SET41 TO** ten thousand, one hundred and ten, which being non-zero, sets the output of Channel 41 to Logic 1.
3. Only constants may be included in the string.

Alternative Form. An alternative form of instructions for consecutive channels is:

3004 SET 41 to 1,0,1,1,0 (the commas are essential)

Channel	41	42	43	44	45
Value	1	0	1	1	0

This form may include calculations, e.g.

3004 SET 41 TO C1<C2,C3<C4,C5<C6,C7<C8

If C1 is less than C2, the expression C!<C" is TRUE, but if C1 is greater or equal to C2 the expression is FALSE. A TRUE result produces a logic 1, a FALSE result produces a Logic 0. The same rules apply to the other three comparisons and the outputs will be set accordingly.

Indexing in a **FOR...NEXT** loop can also be used, as shown below:

3004 FOR I=1 TO 10

3005 SET (40+1) TO SC(I)>LIMIT(I)

3006 NEXT

5.6 OUTPUT FORMAT DECLARATION

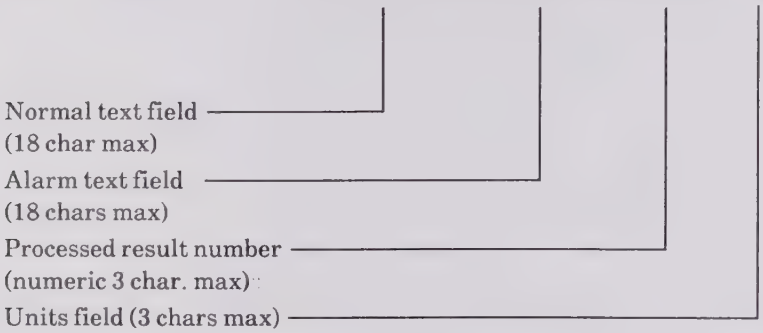
As outlined in section 3, user defined variables and all indexed variables should be defined at the start of the program, and take the general form:

Line number variable;size

variable is the variable name
size is the number of elements in the array
; the semicolon separator is mandatory

However, the declaration can be extended to contain all or part of the additional information that could normally be found in a Full Format result output. This extended variable definition is referred to as an Output Format Declaration and takes the following general form.

Line Number variable;size;"normal text";"alarm text";result no;"units"



Note: Semicolon separators must be included between fields, and all text for printing must be enclosed in inverted commas.

The complete declaration can be reduced to the following combination of items:

a) Processed result number
b) Text
c) Processed result number and units
d) Text and processed result number
e) Text, processed result number and units.

5.6.1 Text

The text included in the declaration will apply to all elements of the array. However by using a PRINT statement with extensions the Output Format Declaration for all the elements in the array can be overridden.

Note: If "normal text" is to be omitted but "alarm" text is to be entered, the inverted commas must still be included as shown below:

Tnnn variable;size;"","alarm text"

5.7 RESULT NUMBERING

Output results can be numbered.

a) Results obtained from the pre-defined variables C1 to C600, whether processed or not, are **always** printed with the relevant variable number. For example, the result from C1 always appears in the following form, followed by a full stop, if the result has been processed.

C001 21.43 RPM

The user cannot influence this numbering, it is controlled completely by the SI3535.

b) Results output from user-created variables always appear **after** the last of the results from the pre-defined variables, accompanied by a "P" number. For example:

P001 23.2 DEG

P-numbers can be allocated within the range P001 to P999. They are under user-control, and can be defined for all user-created variables by a **PRINT** Statement Extension. Also, for indexed variables only, they can be defined in an Output Format Declaration. Whichever method is used for definition, once the first number has been declared for an indexed variable, the rest of the result numbers for that array will be allocated in sequence. eg:

POWER;5;101

This generates five lines of output, numbered P101 to P105 inclusive. If no processed result number is declared, the processor will assign the default value of 1 to the first result, **ie, P001**, and continue sequentially for subsequent results **in that array** with P002, P003, etc. However, if there are several variables declared at the start of the program, with undefined result numbers, the five **POWER** results will be numbered P001 to P005, and the five **LEVEL** and five **TEMP** results will also be numbered P001 to P005.

eg 1000 POWER;5

1001 LEVEL;5

1002 TEMP;5 etc.

Therefore the user should write

1000 POWER;5;1 to number from P001 to P005

1001 LEVEL;5;6 to number from P006 to P010

1002 TEMP;5;11 to number from P011 to P015 etc.

5.8 COMPARISON OF PRINT STATEMENT EXTENSIONS AND OUTPUT FORMAT DECLARATION

The Output Format Declaration is applicable **only** to indexed variables. However, to output results from an indexed variable, the program must also contain a **PRINT** Statement. Hence, if it is required to add text and units to these results, and also to influence the processed result numbers, it is possible to use the Output Format Declaration, a Print Statement Extension, or both, as follows:

a) Declare the full Output Format, ie: **variable;size;"normal text";"alarm text";result no;"units"**, and augment the Print Statement to force Alarm status and select a different output device, if required.

b) Declare the full Output Format and use the Print Statement as for "a", but also use the Extensions to alter the **"text"**, **result number** and **"units"** for specific processed results from each array element.

c) Declare the indexed variables **name** and **size** and use the Print Statement with Extensions to define the rest of the information for each array element.

The Print Statement Extensions take precedence over the Output Format Declaration. When the processor encounters a Print Statement involving an indexed variable in the program, it first checks if there are any Extensions to be actioned. If there are not, it checks if there is an Output Format Declaration. Failing both these, the processor outputs the results, with no text or units, renumbering them from P001 upwards.

6 MATHEMATICAL OPERATORS AND FUNCTIONS

6.1 MATHEMATICAL OPERATORS AND PRIORITIES

The following groups of operators are available for processing SI3535 data, using the symbols given in Table 4.2.

Table 4.2 *Mathematical Operators and Functions*

TYPE	SYMBOL	OPERATION	EXAMPLE
Mathematical	+	Addition	A+B
	-	Subtraction	A-B
	*	Multiplication	A*B
	/	Division	A/B
	**	Integer Exponentiation	for A^2 type A**2; the exponent must be an integer
	()	Parentheses	

The operations are performed according to standard mathematical rules, as follows:

1st Priority: Parentheses
2nd Priority: Integer Exponentiation
3rd Priority: Division
4th Priority: Multiplication
5th Priority: Addition
6th Priority: Subtraction

6.1.1 Pending Operations

Expressions containing a maximum of 4 pending operations can be entered. The expressions are evaluated from left to right, and as the processor find higher priority calculations to perform, the lower priority ones are left pending.

6.1.2 Parentheses

Parentheses can be nested up to 4 levels,
eg. A=LN(SQR((5*(C4+2.3))/13))

1st LEVEL = $5\times(C4+2.3)$

3rd LEVEL = $\sqrt{\left(\frac{5\times(C4+2.3)}{13}\right)}$

2nd LEVEL = $\frac{5\times(C4+2.3)}{13}$

4th LEVEL = $\log_e \sqrt{\left(\frac{5\times(C4+2.3)}{13}\right)}$

6.2 TRIGONOMETRIC AND EXPONENTIAL FUNCTIONS

The SI3535 trigonometric and exponential functions are summarized in Table 4.3.

Table 4.3 *Trigonometric and Exponential Functions*

FUNCTION	OPERATION	EXAMPLE
ABS(X)	Absolute value of x	+6.6 becomes 6.6 - 6.6 becomes 6.6
SQR(X)	Square root of x	$\sqrt{x}$, x must be positive
SIN(X) COS(X)	Sine and Cosine trigonometric functions	x must be in radians
ATAN(X)	Arctan inverse trigonometric functions	x must be in radians
EXP(X)	Exponential to the power x	e_x, x must be an integer
LN(X)	Natural logarithm of x (i.e. base e)	Log_e x
INT(X)	Integer part of x	+6.6 becomes 6.6 - 6.6 becomes 6.6 The SI3535 takes the first integer less than the value given. Since mathematical negative numbers are less than zero, -7 is less than - 6

These functions are written into the program statements in the following format:

4020 RESULT = SQR(C6)

When this is printed out, the value stored in the variable RESULT will be the square root of the result from Channel 6, as measured in task 4.

A more complex function,

4020 RESULT = SQR(C6*7)

provides the square root of (7 times the result from Channel 6). This complexity can be extended up to 4 levels of expression nesting in any one statement. Expressions requiring more than 4 nesting levels must be divided into two or more statements.

6.3 RELATIONAL AND BOOLEAN OPERATORS

The relational and Boolean operators that can be included in a SI3535 processing program are summarized in Table 4.4.

Table 4.4 *Relational and Boolean Operators*

FUNCTION	SYMBOL	OPERATION	EXAMPLE
RELATIONAL	<	Less than	A<B
	>	Greater than	A>B
	=	Equality	A=B
	<>	Non equality	A<>B
	<=	Less than or equal to	A<=B
	>=	Greater than or equal to	A>=B
BOOLEAN	AND	Boolean and	A AND B=C
	OR	Boolean or	B=C OR A=C

Combinations of these operators can be used in statements of the type:

IF A>P AND B>Q THEN...

IF A>P OR B>Q THEN...

7 ENTERING AND EDITING PROGRAMS

A program can only be input to the SI3535 when no tasks are running. If a task is running the HALT command (HA) must be sent before any program lines will be accepted. The task-coded 4-digit line number indicates to the SI3535 that the input should be interpreted as a program line; lines not starting with a line number are interpreted as a SI3535 command string.

7.1 ENTERING PROGRAMS

Programs are entered by typing each statement line, starting with a line number, and ending with a carriage return C/R. Use is made of the processing keys and keypad characters. Front panel users should use the screen editor (see User Guide). Remote entry is done with

Txxx <Program Line> C/R

Each task created within the SI3535 has a fixed allocation of line numbers for the program associated with that task.

ie **T000 to T999**

T is the task number in the range 1 to 8.

Program lines can be corrected by typing the replacement line in full, including the line number, followed by C/R. A line may be deleted from a program by typing the line only, followed by C/R. To insert a line between two adjacent lines (line numbers permitting) type a suitable line number, followed by the new program statement and C/R. If an appropriate line number is not available, the program can be re-numbered (Section 5.7).

A program can be completely or partially re-numbered using the Renumbering (RE) command. The format of the Renumbering Command is as follows:

RE Old line number , New line number , Increment between.

7.2 SINGLE AND MULTI-STATEMENT LINES

Program lines consist of one or more statements; when memory size for programs is not a problem, there are advantages, as follows, in assigning a new line to each statement:

a) The clarity and presentation of the program is improved.

b) Line length will not exceed the display capacity.

c) Program maintenance and modification will be simplified.

For large programs, and more efficient utilization of memory capacity, it may be necessary to use multi-statement lines. (Each program line always occupies 200 bytes of memory, irrespective of its length).

Note: In practice the actual number of characters that a line can contain is dependent on the number and type of statements included in that line. There is internal expansion of the line, when coded for transfer to memory; the expansion factor varies according to the line content.

An example of a multi-statement line is given below:

1050	IF C1 < > 10 THEN:	GOTO 1100:	ENDIF
Line Number	Statement 1	Statement 2	Statement 3

Brief details of statement control words and expressions follow:

7.2.1 Control Words

The statement control words are of four types:

a) Program control, eg **IF...THEN, GOTO, STEP**

b) Data Output control, eg. **PRINT, WRITE,KILL**

c) Output control, **SET TO**

d) Task control eg. **TRIGGER, ABORT**

7.2.2 Expression

The expression component of a statement may consist of one or a number of the following components parts.

a) Variable-simple or indexed, eg. MAX, TOTAL(I)

b) Constant-numeric, exponential, textual,
eg. 1.5, 4.375E-5,"OVERLOAD".

c) Operator-mathematical, relation, eg. +,-,>.

d) Function-Mathematical functions, ef. ABS(x),SIN(x).

7.3 STATEMENT SEPARATORS/DELIMITERS

Where multi-statement lines are used, colons must be inserted as separators between each pair of statements. The main exception to this rule are statements of the type **IF...THEN**, where the word **THEN** acts as an automatic statement separator. Also no separator is needed after the final statement in a line, as it is automatically completed by pressing **ENTER**.

However, for consistency, a colon can be inserted after every statement; it will be ignored if it is not required.

7.4 EXPRESSIONS AND ASSIGNMENTS

The term "expression" is defined here as follows:

A combination of variable and function names and constants, which are related to each other via an operator, (as defined in Section 3).

In its most elementary form, an expression may be simply a constant, eg. 2, or a variable, eg. TEMP. However, the more usual form includes an operator, such as **TEMP*8**, (ie. TEMP multiplied by 8), **C1>5**, and **(C1 + C2)/C2**.

In **BASIC**, from which the SI3535 software language is derived, the "=" sign has two distinct uses:

a) In "assignments" it has the following meaning:

`the expression to the left of the "=" sign is assigned the value of the expression to the right.'

A simple example is A=8, where the variable A is assigned the value of the constant 8.

b) In equations it has the following meaning:

'the expression to the left of the "=" sign is being tested for equality with the expression on the right.'

This meaning is usually encountered in **IF...THEN** statements; a typical application would be:

IF C1 = C2 THEN proceed accordingly.

The processor deduces the appropriate meaning of the "=" sign from the context of the lines in which it appears.

The programmer can write either **A=C1*C2** or **C1=C1*C2** both are equally valid when it is remembered that they are assignments and not equations. The distinction to be made here is that A is a user-created variable whereas C1 is pre-defined variable.

7.5 PROCESSING KEYS (Not via remote interface)

Eight dual-purpose PROCESSING softkeys provide the words most commonly required for programs. On the SI3535D, the softkeys are accessible by pressing the PROGRAM MENU key. The other, less-common statements etc. that the logger can recognise are constructed character by character from the keypad.

Some of the words are always grouped together in control structures, while others are employed singly, such as PRINT.

7.6 EDITING KEYS (Not via remote interface)

All keys are available when writing or editing processing programs, and all except LINE INSERT can be used when setting up the various menus.

Each edit key has an auto-repeat facility that if the key is held pressed, the relevant action is repeated. For example, characters will continue to be deleted, until either the key is released, or there are no characters left in the line).

To edit a processing program which is not currently being displayed, first press PROGRAM MENU, followed by PROCESS. Enter the number of the line which is to be edited, then use the EDITOR keys as described in the following sections. On completion of editing, pressing EXIT will produce the displayed state "Ready for command". Alternatively, the user may simply press the key for the next required function, e.g. **RUN**.

↑ STEP BACK	These keys enable the user to step through
↓ STEP FORWARD	processing programs line by line in either direction, for inspection and/or alteration. Branches of menus can be similarly viewed in most cases. If the sequence is already displaying the final permissible line, in either direction, any attempt to step further in that direction sounds the Error Tone.
LINE INSERT	This facility is only available processing programs.
LINE DEL	To delete a line, operate the step back/forward keys until the line to be deleted appears in the display. Press LINE DEL, the original line disappears and the remaining lines automatically close up so that the next one in the sequence now appears in the display. With processing programs, the line numbering is automatically adjusted.

7.7 RESEQUENCING

The resequencing facility is for use with programs which have been entered via a remote interface, and require editing at the front panel of a SI3535D. For programs initially written on non-consecutive lines, the processor can be instructed to resequence line numbers to be consecutive. In order to perform resequencing, the SI3535 is set up for **READY FOR COMMAND** prompt, and a line is entered of the form:

RE Old First Line Number , New First Line Number , Step Size

The processor automatically initiates resequencing when the **LINE INSERT** or **LINE DELETE** functions are used. A failed resequence is indicated by Error Code 161.

a) Line 999 or a specific task No. is in use, and **LINE INSERT** is used to add an extra line.

b) Line 999 of a specific task No. would be exceeded by the resequencing action.

c) An attempt is made to transfer a program from one task to another, eg. from task 3 to task 8 by:

RE 3000,8000,1

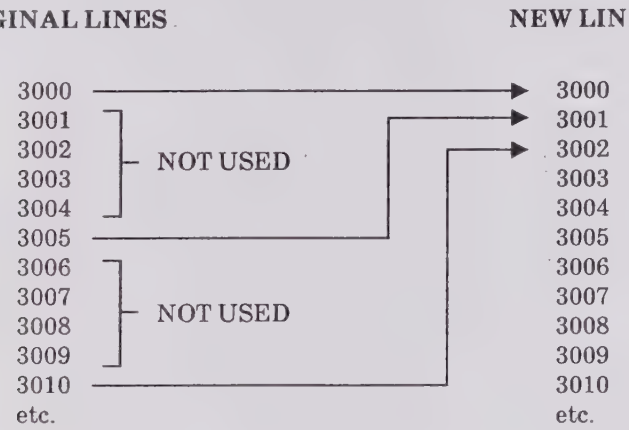
which is not allowed.

An example of resequencing is given below:

The original program started on line 3000, using every fifth line number. Having completed the program, it is necessary to eliminate all the gaps, i.e. have the program on consecutive lines, but still starting on line 3000. This would be keyed in as:

RE 3000,3000,1 ENTER

The line numbers will now be re-allocated as shown below:



The processor automatically initiates resequence whenever the **LINE INSERT** or **LINE DELETE** functions are used but not from the remote interface.

8 DIGITAL INPUTS

Processing can be performed on digital signals, eiher as single bit inputs or as groups.

8.1 SINGLE BIT INPUTS

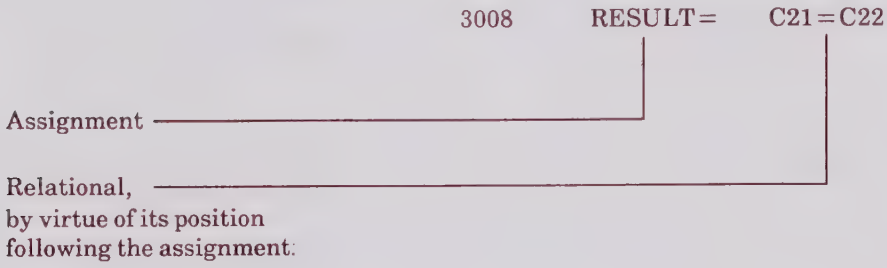
Each single bit input is regarded as being either a Logic 1 or a Logic 0. The two Boolean operators **AND** and **OR** can be used in the **IF...THEN** statements (Section 4).

eg: **3005 IF C21 = 1 THEN....**

3005 IF C21 = 1 AND C22 = 1 THEN....

3005 IF C21 = 1 OR C22 = 0 THEN....

The next example of single bit input processing includes an application of the dual use of the "=" sign (Section 7.4) "Assignments"). Consider the following statement, where C21 and C22 are Status inputs, monitoring switch closures in the external plant.



a) When the plant switch is closed, a logic 0 state is detected.

b) When the plant switch is open, a logic 1 state is detected.

Hence, if both switches are closed, C21 and C22 are both at logic 0, whereas if both switches are open, C21 and C22 are both at Logic 1. In either case the equation is TRUE, C21 equals C22, which yeilds a logic 1 as the result. Both cases therefore simplify the original line to:

RESULT = 1

(ie. the processor assigns the value "logic 1" to the variable RESULT)

Conversely, when one switch is open and the other closed, C21 cannot equal C22. The equation is FALSE and a logic 0 is produced as the result. The original line now reduces to:

RESULT = 0

and the processor assigns the value "logic 0" to the variable RESULT.

Another example is:

CONDMET = C21 AND C22 AND C23 = 0

As before, the equation can be either TRUE or FALSE, and hence CONDMET will be assigned either a 1 or 0.

This line of program would probably be followed by THEN,

eg: **IF** CONDMET = 1 **THEN** etc

8.2 GROUPED INPUTS

Digital inputs can be grouped together in the Status/Event Channel Definition; the grouping process causes these inputs to be regarded as a decimal number. For example, suppose that the inputs of Channels 21 to 24 are to be grouped for processing. Channel 1 only must be declared in the Channel Definition. When the Sensor menu offers "Select digital input function", press the softkey "bit" and enter the number "4".

The required input condition could be:

C24	C23	C22	C21
1	1	0	1

The grouping process causes these inputs to be regarded as a binary number, in this case:

8	4	2	1
1	1	0	1

which equals 13 (ie. 8 + 4 + 1)

The programmer can now write:

IF C21 = 13 **THEN** proceed accordingly.

This method is much more compact than the alternative of testing for four single bits, which would be written:

IF C21 = 1 **AND** C22 = 0 **AND** C23 = 1 **AND** C24 = 1 **THEN** etc.

9 TASK CROSS CONTROL

By using cross-triggering and cross-aborting statements within the program for a specific task, it is possible to control the activity of other tasks. These statements can be included within **IF...THEN** control structures.

9.1 TRIGGERING OTHER TASKS

The following statement causes Tasks x, y and z to be triggered:

TRIGGER x,y,z ...

If the task is already active, the trigger will simply be ignored. Although up to all eight tasks can be specified in the task list, self-triggering within the same program is of little practical use.

9.2 ABORTING OTHER TASKS

The following statement causes Tasks p,q and r to be aborted (ie. stopped).

ABORT p,q,r ...

All eight tasks can be included in the task list if required, and self-aborting within the same task program is acceptable.

Note: Due to internal software time delays, and task prioritization, it is possible for task cross-control statements to deviate from their apparent sequence.

eg. Task A can issue a statement to **TRIGGER** Task B, chronologically after a statement to **ABORT** Task B has been issued by Task C. However, due to these internal time delays, the **TRIGGER** is not implemented, since it is instantaneously followed by the **ABORT** statement.

10 COUNTERS AND TIMERS

The SI3535 includes 8 Counters and 9 Timers, which although defined as global indexed variables, are handled by the processor as special cases. Although the general indexed form must include parentheses (eg, **COUNT(1), TIMER**(1)) the use of parentheses around individual elements is optional; hence **COUNT4** and **COUNT(4)** are both equally valid.

10.1 COUNTERS: COUNT1 TO COUNT8

Each task has its own dedicated Counter, which can be interrogated, but not modified by the processing program. Each Counter produces a total of the number of scans performed in that task since it was triggered; Counters are automatically set to zero by the **RUN** command.

NOTE: Counters cannot be preset to an initial count; the maximum count 2^{31} is displayed as 2.147E9.

The functions available are as follows:

COUNT1 to **COUNT8** inclusive, which are Scan Counters for Tasks 1 to 8.

COUNT(1) which affords indexed access to Counters 1 to 8 inclusive.

To determine the total held in any of the Counters since RUN was operated, eg in Counter No. 7, simply write:

```
Tnnn   PRINT COUNT7
```

An example of the use of a Counter in the calculation of a running average of Channels 1 to 10 in Task 6, over several scans, is given below:

```
6007   FOR I = TO 10
6008   SUM(I) = SUM(I) + C(I)
6009   AVERAGE(I) = SUM(I) COUNT6
6010   PRINT AVERAGE(I)
6011   NEXT
```

Counts are updated at the start of a scan. If a second scan has started before the first has been fully processed (due to short scan interval), the COUNT reading indicates the number of scans started but not necessarily completed.

For each identification, the sign of the COUNT reading goes -ve when the number of scans requested in the Task Definition is reached. This also occurs if the Task is aborted. However, if the Task is aborted and then re-triggered, the sign will revert to positive, and the count will continue from the total reached prior to the abort command.

10.2 TIMERS : TIMER0 TO TIMER8

The 9 elapsed timer variables, **TIMER0** to **TIMER8** are not task dependent, and can be used in any task. The functions available are as follows:

TIMER0 automatically records time elasped since RUN was operated, for the whole of the SI3535 and is resettable.

TIMER1 to **TIMER8** inclusive - timers can be allocated to any task, or all eight to the same Task if required.

The nine Timers operate as follows:

TIMER0 is reset to zero when **RUN** is pressed, and also by executing any Statement which assigns a value to it, the most logical being **TIMER0 = 0**. The Timer would also be reset by eg **TIMER0 = 42**, but note that the value 42 is not preset into it. As soon as the Timer has been reset it automatically starts again from zero.

TIMERS 1 to 8 are started by executing any statement which assigns a value to them, they are not started by RUN or by the start-up of any Task which uses them. To start eg **TIMER5** write **TIMER5 = 0**. As with **TIMER0**, if **TIMER5 = 42** is written, **TIMER5** will be reset to zero, not preset to 42.

All the Timers can be interrogated as often as required eg:

```
7009    IF TIMER0 > 10.5 THEN
7010    PRINT "TIME EXCEEDED"
7011    ENDIF
```

TIMER(I) provides indexed access to **TIMERS0 to 8** inclusive.

The Timers count in seconds, with a resolution of 1/10th of a second, to a maximum of 2^{31}, which represents approximately 6.8 years! If a print-out of a long time period is required, the processing facility can be used to convert the result into hours, minutes and seconds. Normally, however, hours and minutes information would be taken from the Start Task and Done Task Markers.

11 RANDOM CHANNEL ACCESS

Generally, for tasks in which channels are scanned consecutively, channel variables of the form C1 or as indexed variables C(I) are satisfactory. However, in some tasks the scan set may have been defined in a non-consecutive order, or the actual channels in the scan set may vary. For tasks of this nature, use should be made of the Scanned Channel variable, for simplification of the processing program. This is a local indexed variable which gives access to the Ith Channel in a set of Scanned Channels, only associated with a specific task:

SC(I) Ith Channel result in scan set

SH(I) High Limit for Ith channel in scan set

SL(I) Low Limit for Ith channel in scan set

SN(I) Gives Channel number of Ith channel scanned

Assuming channels to be processed have been scanned in an irregular sequence, such as 8, 14, 30, 64-69, an application of the Scanned Channel facility is to print the results from this sequence. The program is as follows:

```
1000    FOR I=1 TO 9
1001    PRINT SC(I)
1002    NEXT
```

NOTE: I=1 TO 9 not 8 TO 69

The **FOR....NEXT** loop performs normally with the exception that in line 1001, when I=1, it is not Channel 1 that is printed but the first Channel in the scan set (ie Channel 8). Similarly, when I=2, the second Channel, 14, is printed etc. (in general terms, it is the Ith Channel in the scan that is accessed).

All other **FOR...NEXT** loop facilities still apply, eg it is still permissible to write: **FOR I=2 TO 4** in which case only the 2nd, 3rd and 4th Channels will be accessed; STEP can still be used, as in:

```
FOR I=1 TO 9 STEP 2
```

in which case the 1st, 3rd, 5th, 7th and 9th Channels are accessed.

11.1 SCANNED CHANNEL VARIABLE - APPLICATION

An example of a program using the Scanned Channel Variable facility is provided. Assume that Task 1 scans the five non-consecutive Channels 5, 9, 11, 17, 19. A program to record the cumulative total value of each Channel, over several scans, is shown :

```
1000 TOTAL;5

1001 FOR I=1 TO 5

1002 TOTAL(I)=TOTAL(I)+SC(I)

1003 PRINT TOTAL(I)

1004 NEXT
```

Pressing RUN to start scanning reads the declaration in line 1000. This line instructs the processor to allocate storage space for an indexed variable called **TOTAL** which consists of five elements forming the variables **TOTAL(1)** to **TOTAL(5)**. These five variables all have an initial value of zero, and remain in existence until **RUN** is operated again to stop scanning. The rest of the program is executed once for each time that Task 1 causes the Channels to be scanned, but line 1000 is not actioned again during these repetitions.

The program continues with a **FOR...NEXT** loop, which contains a novel feature in line 1002. Remembering that **SC(I)** refers to the Ith Channel rather than to the Channel number itself, line 1002 is effectively expanded by the processor to mean:

```
TOTAL(1)=TOTAL(1)+1ST SCANNED CHANNEL

TOTAL(2)=TOTAL(2)+2ND SCANNED CHANNEL

TOTAL(3)=TOTAL(3)+3RD SCANNED CHANNEL

TOTAL(4)=TOTAL(4)+4TH SCANNED CHANNEL

TOTAL(5)=TOTAL(5)+5TH SCANNED CHANNEL
```

Reading the first line as an example, from left to right, it means:

"the value to be stored in **TOTAL(1)** is to be the value already in there plus the current value of the first scanned Channel, ie Channel 5".

On the first scan, the value already in **TOTAL(2)** will contain the first value measured on Channel 9, and so on for the other three Channels

After the second scan, the five variables will contain the sum of the values measured on their respective Channels over the first two scans. The summation continues for as many scans as the Task commands.

Line 1003 causes the values of the five variables to be printed out after every scan. These results can have additional information included, either by using the Print Statement Extensions or the Output Format Declaration. The latter could be:

```
1000 TOTAL;5;"TOTAL FLOW";"HIGH FLOW";1;"LTR"
```

where LTR represents litres.

12 STORING AND REPLAYING PROCESSED DATA AND PROGRAMS

For versions SI3535D and SI3535G the integral disk drive allows a maximum of 4 programs to be stored on floppy disk in MS DOS format and maximum of 3 programs to be stored on floppy disk in HPLIF format. Each program effectively consists of Channel and Task Definitions, together with processing programs included in the Tasks.

The data will be written as 8 bit ASCII characters in one of three formats: Full, Compact or Coded Binary. The number of measurements saved in these formats differ between HP-LIF and MSDOS.

Compact and Full Format: Disks recorded in Compact or Full Format can be replayed in any suitable read station without the need for any decoding. For HP-LIF, one disk can accommodate a maximum of 28,625 measurements in Compact form or 14,995 measurements in Full format. The equivalent values for MSDOS are 33,175 measurements in Compact form or 17,375 measurements in Full form.

Coded Binary Format: If SCORPIO processing has not been used, each 10 character ASCII string will contain numeric data in binary form, encoded into printable ASCII characters. Up to 52,480 results in HP-LIF and 60,820 results in MSDOS can be accommodated in this form.

When **processed results** are in Coded format the following conditions are applicable:

a) If the results are in the pre-defined variables C1 to C600, each result constitutes only one record on the disk.

b) If the results are the derived processed type, from user-created variables, (eg. A,TEMP,DEPTH(I) etc) then each one is automatically accompanied by a "POINTER" record to assist decoding, hence:

   (i) The disk now accomodates fewer results.

   (ii) When decoded processed data is being replayed to the SI3535 from the disk, **the original processing program must be present in memory, to enable the POINTERS to be decoded.**

Disks recorded in Coded binary format would normally be replayed in SCORPIO, but can be replayed in other read stations if suitable decoding is applied to the data. However, if the disk contains processed data the records containing pointer information should be ignored as they contain no meaningful information for the user. The pointer record may be identified by a channel number of zero. Further details on output formats are given in Chapter 5 : Output Formats.

13 MEMORY CAPACITY

The amount of memory available for user processing programs depends on the number of data acquisition modules fitted to the SI3535, and on the size of any programs already stored. Each occupied module slot from 4 to 13 in a SI3535 unit, or slave expansion unit (SI3535E), is equivalent to 520 bytes of memory.

13.1 PROCESSING PROGRAM MEMORY REQUIREMENTS

The amount of memory required by a program depends on both the number of statement lines, the number of user-defined variables and elements in each array.

In the unlikely event that your program occupies too much memory, codes 151 or 152 will be displayed informing you to reduce the program length. If memory does become limited multi-statement lines can be used, since each program line occupies 200 bytes of memory, regardless of the number of statements.

14 PROCESSING PROGRAMMING TECHNIQUES

The following information is intended to assist the user when writing Processing programs for the SI3535. Important points, which have already been mentioned in this Chapter, are re-emphasised:

1. Always plan the processing program away from the device, whether it is to be entered from the SI3535 front panel or via a remote interface.

2. Start line numbering from line T000, where T is the task number. Use consecutive line numbers, and avoid the use of line numbers not followed by a declaration or statement.

3. Declare indexed variables at the start of the program, and use descriptive names for each variables.

   Note: Variables are always initialized, cleared to zero, when RUN is given. However, if a program is aborted and then retriggered the variables will not be re-initialized.

4. Declare indexed variables that are to be accessed by more than one task in the program for the lowest numbered task in which they are used.

   Note: Indexed variables must only be defined once.

5. Suffix commonly used variables, eg I,J etc., with task numbers, eg I1, J2. This should avoid accidental use of the same variable name in more than one task.

6. To make the program easy to understand and/or update, use single statement lines, when program space allows. If program space is critical use multi-statement lines.

7. Use `first scan processing', by devoting a small section at the start of the program to actions that are only to be performed the first time that the program is actioned, processing time can be reduced. Use first scan processing to perform the following:

   a) Assign initial values to simple variables and elements of indexed variables.

   b) Perform any once only calculations.

8. Use the **ELSE** form of the **IF...THEN** statement whenever meaningful. To increase effectiveness of **IF...THEN** statements, use with BOOLEAN expressions.

   eg **IF C1>C2 AND C3<C5 THEN**
   .....
   ELSE

9. Manipulate data using the channel variables C(I) and SC(I) rather than introducing temporary variables; this saves on program memory.

10. Use the scan channel variables SC, SH, SL and SN to access channel or alarm data, in preference to the channel variable, C, and alarm variables H and L. This simplifies external adjustment of channel sets, ie the actual channel, numbers are irrelevant, providing the number of channels in the channel set does not change.

11. Use the **KILL** statement, when channel data is no longer required for processing or output, to free buffer space; buffer space will be freed immediately the KILL command is actioned. In addition to freeing buffer space post processing will be speeded up, as alarm and checking and post processing is not carried out on channels that have been killed.

12. Save memory by overwriting existing variables instead of creating new ones.

13. Economise on statements by including expressions in **PRINT** and **SET** statements.

14. Use the channel set alarm limits to save assignments and variable space. Where possible use the SI3535 alarm checking and conversion facility since this is faster than using the processing program.

15. Avoid programs that generate a large number of processed ouput, ie using the **PRINT** statement, which could cause buffer exhaustion.

16. Avoid use of the **WRITE** statement for standard task output, use **WRITE** for the following:

    a) Sending messages to other devices.

    b) Alarm messages

    c) Producing information in tabular form.

17. Avoid use of GOTO.

18. Do not attempt accurate use of the timers. If a timer is to be used store the time to `freeze' it before it is used in calculations etc.

19. Do not use the task counters for calculation. Keep a count within the processing program; it is more likely to be accurate, as the processing time can overlap into the next scan.

15 EXAMPLE PROGRAMS

15.1 CONTENTS

The following example programs are given in this section :

1. Single Channel Average over 5 readings.
2. Average of 3 channels on every scan, with a cumulative average of each channel every five scans.
3. Maximum and minimum reading.
4. Output on change
5. Output on change, giving old and new values and the difference between them.
6. Warning if channel reading is outside limits.
7. Alarm threshold.
8. Walking window average on one channel.
9. Walking window average on three channels.
10. Mean and standard deviation.

15.2 INTRODUCTION

For simplicity, most of the programs are shown using a minimum number of channels. However, once the basic principles have been understood, the programs can be extended to encompass many more channels and to include additional statements. As with any software language, there are probably several ways to achieve the desired result; the examples given are suggestions only.

Every channel whose result is to be processed must be "configured" in the channel definition. If this is not done, the channel listing will appear as:

```
CHANNEL  001
measurement UNDEFINE
D
conversion 0
alarm OFF
o/p 000,000
analogue o/p  0
text 0
alarm text  0
```

All the examples in this section use a minimum channel definition of the form :

```
----------------------
CHANNEL  001
measurement Volts
 DC
conversion 0
alarm OFF
o/p 000,000
analogue o/p  0
text 0
alarm text  0
```

Each is accompanied by a typical print-out, and also the task definition which was set up to produce it. This definition may be amended, if so desired, e.g. to include different channels, perform more scans, output all results rather than just processed ones etc.

The programs are written, for clarity, on a one statement per line basis. However, in practice, they could be considerably condensed by using multi-statement lines.

15.3 SINGLE CHANNEL AVERAGE OVER 5 READINGS

TASK DEFINITION

```
TASK  001
title TIME AVERAGE 1

Function Scan
trigger Timer
delay 00-00:00:00.0
scans 5
scan Inter
interval 00-00:00:01
.0
channels 001
rate 10/s
processing On
data to be logged Pr
oc
header LogData
output device Print
Format Full
tasks triggered 0
tasks aborted 0
```

PROGRAM

```
1000 SUM=SUM+C1
1001 COUNTER=COUNTER
+1
1002 IF COUNTER=5 TH
EN
1003 AVERAGE=SUM/5
1004 PRINT AVERAGE:"
TIME AVERAGE";1;"VDC
"
1005 COUNTER=0
1006 SUM=0
1007 ENDIF
```

PRINT-OUT

```
RUN   16:44:19 06-0
 TIME AVERAGE 1
S T 1 16:44:23.7
 :TIME AVERAGE
P 001 4.23186  VDC

D T 1

HALT  16:44:34 06-0
```

PROGRAM KEY

1 All variables start with value zero when RUN is pressed to start scan

2 On the first scan, SUM stores the value measured on Channel 1. On the second scan, SUM retains the previous value, and adds to it the new value from Channel 1. This process continues for all the 5 scans specified in the Task Definition. SUM then holds the sum of all five readings on Channel 1.

3 On the first scan, COUNTER stores the value "1". On the second scan, COUNTER stores the value 1 + 1, i.e. 2.
This process continues for all 5 scans, until COUNTER holds the value 5.

4 On the first 4 scans, COUNTER cannot contain 5, therefore the processor ignores all intervening lines and jumps straight to the ENDIF in line 1007. On the 5th scan, COUNTER will contain 5, so the rest of the program will be actioned.

5 The total value stored in SUM is divided by 5, and the result, i.e. the required average value, is stored in the new variable AVERAGE.

6 On the 5th scan only, the value in AVERAGE is output, in this case to the Printer, as specified in the Task Definition.

7 The contents of the two variables is set to zero. Strictly speaking, there is no need to do this, as the Task is limited to 5 scans. To obtain the average value from a further 5 scans, RUN would have to be pressed again to halt the logger, and again to re-start, so the variables would automatically have zero value. However, it is good practice to clear variables, once they have served their purpose. Note that any reading stored in AVERAGE will be cleared automatically whenever a new value is assigned to it.

8 Title from Task Definition

9 Title from line 1004 of program

10 Processed result, which is the required average reading from Channel 1 over 5 scans.

15.4 AVERAGE OF 3 CHANNELS ON EVERY SCAN, WIH A CUMULATIVE AVERAGE OF EACH CHANNEL EVERY 5 SCANS

TASK DEFINITION

```
   TASK  001
   title GROUP/TIME AVE
   RAGE
   Function Scan
   trigger Timer
   delay 00-00:00:00.0
   scans *
   scan Inter
   interval 00-00:00:02
   .0
1  channels 005,011,018

   rate 10/s
   processing On
   data to be logged Pr
   oc
   header LogData
   output device Print
   Format Full
   tasks triggered 0
   tasks aborted 0
```

PROGRAM

```
2  1000 TIMESUM(3)
3  1001 GSUM=0
4  1002 FOR I=1 TO 3
   1003 GSUM=GSUM+SC(I)

5  1004 TIMESUM(I)=TIME
   SUM(I)+SC(I)
   1005 NEXT
6  1006 PRINT GSUM/3;"G
   ROUP AVERAGE";I;"VDU
   "
7  1007 TALLY=TALLY+1
8  1008 TOTAL=TOTAL+1
9  1009 IF TALLY=5 THEN

10 1010 FOR I=1 TO 3
11 1011 AVER=TIMESUM(I)
   /TOTAL
   1012 PRINT AVER;"TIM
   E AVERAGE";SN(I);"VD
   U"
   1013 NEXT
   1014 TALLY=0
   1015 ENDIF
```

PRINT OUT

Over 5 scans. Note that first 4 scans are Group Average only; the 5th scan also gives the cumulative averages:

```
RUN    01:42:39 01-01
  GROUP TIME AVERAGE
 T 1 01:42:39.9
 :GROUP AVERAGE
P 001 4.67366  VDU
 T 1
  GROUP TIME AVERAGE
 T 1 01:42:41.9
 :GROUP AVERAGE
P 001 4.67366  VDU
 T 1
  GROUP TIME AVERAGE
 T 1 01:42:43.9
 :GROUP AVERAGE
P 001 4.67366  VDU
 T 1
  GROUP TIME AVERAGE
 T 1 01:42:45.9
 :GROUP AVERAGE
P 001 4.67366  VDU
 T 1
  GROUP TIME AVERAGE
 T 1 01:42:47.9
 :GROUP AVERAGE
P 001 4.67366  VDU
 :TIME AVERAGE
P 005 6.43100  VDU
 :TIME AVERAGE
P 011 2.22600  VDU
 :TIME AVERAGE
P 018 5.33400  VDU

 T 1
```

PROGRAM KEY

1 Note non-consecutive Channel numbers.

2 A 3-element variable, to store results from the Channels for cumulative averages.

3 This variable is assigned the value 0, to zero it for each scan prior to receiving the Group Sum results.

4 After each scan, GSUM holds the sum of the readings from the 3 Channels, i.e. the Group Sum. Note the use of SC(I) to read the non-consecutive Channels 5,11 and 18.

5 TIMESUM(1),(2) and (3) store the readings from the 1st, 2nd and 3rd scanned Channels respectively.

6 Output the Group Sum divided by 3 i.e. the Group Average, to the device selected in the Task Definition, in this case the Printer.

7 Records the groups of 5 scans.

8 Records the total number of scans.

9 Start of an IF . . . THEN loop to output the average reading from each Channel on every 5th scan. The loop is *not* actioned on the first 4 scans; the program skipping straight to ENDIF on line 1015.

10 Note that this is the second time that I has been used as an index in the program. This is good economy, as it minimises the number of variables to be stored. However, I can only be used again here because there is no conflict with the previous application.

11. The FOR . . . NEXT loop is executed 3 times. On the first execution, the sum of the readings stored in TIMESUM(1) is divided by the number of scans stored in TOTAL. The result is stored in the simple variable AVER (for "average"), thus leaving the accumulated sum of the readings undisturbed in TIMESUM(1). The Print Statement causes the value of AVER to be added to the queue of data awaiting outputting, thus releasing AVER ready to accept the average value from TIMESUM(2)/TOTAL on the second execution of the loop. A similar sequence occurs for TIMESUM(3).

Remember that PRINT expression must *never* start with an indexed variable, i.e the temptation to simply write:

PRINT TIMESUM(I)/TOTAL must be firmly resisted!

The use of SN(I) to define processed result number enables the original Channel identity to be preserved, as shown in the results print-out.

15.5 MAXIMUM AND MINIMUM READINGS, ON ONE CHANNEL, OVER 10 SCANS

TASK DEFINITION

```
TASK  001
title MAX/MIN

Function Scan
trigger Timer
delay 00-00:00:00.0
scans 100
scan Inter
interval 00-00:00:02
.0
channels 001
rate 10/s
processing On
data to be logged Pr
oc
header LogData
output device Print
Format Full
tasks triggered 0
tasks aborted 0
```

PROGRAM

```
  1000 IF N=0 THEN
  1001 MAX= -100000
1 1002 MIN=100000
  1003 ENDIF
2 1004 N=N+1
3 1005 IF C1>MAX THEN
    MAX=C1
  1006 ENDIF
4 1007 IF C1<MIN THEN
    MIN=C1
  1008 ENDIF
  1009 IF N=10 THEN
  1010 PRINT MIN;"MIN
  VALUE";1;"VDC"
5 1011 PRINT MAX;"MAX
  VALUE";2;"VDC"
  1012 N=0
  1013 ENDIF
```

PRINT-OUT

Over 30 scans.

```
RUN    16:30:41 07-03
  MAX/MIN
S T 1 16:30:59.4
 :MIN VALUE
P 001 0.0364200VDC .
 :MAX VALUE
P 002 0.153280 VDC .

D T 1
RUN    16:33:22 07-03
  MAX/MIN
S T 1 16:33:40.8
 :MIN VALUE
P 001 0.096510 VDC .
 :MAX VALUE
P 002 0.284130 VDC .

D T 1
RUN    16:35:24 07-03
  MAX/MIN
S T 1 16:35:42.9
 :MIN VALUE
P 001 0.759320 VDC .
 :MAX VALUE
P 002 1.52168  VDC .

D T 1
```

PROGRAM KEY

1 This IF . . . THEN statement is executed only on the first scan after RUN has been pressed. The effect is to pre-set the variable MAX to a very small value, and MIN to a very large value. This gives the processor two initial values against which to compare the first readings.

2 The variable N acts as a tally counter, incrementing by 1 on every scan.

3 Whenever the reading on Channel 1 is greater than that already stored in MAX, the latter is replaced by the new reading.

4 Whenever the reading on Channel 1 is less than that already stored in MIN, the latter is replaced by the new reading.

5 This IF . . . THEN statement is actioned only on the 10th scan. The minimum and maximum values, stored in MIN and MAX respectively, are output to the device specified in the Task Definition, in this case the Printer.

N is reset to zero, ready for the next 10 scans.

15.6 OUTPUT ON CHANGE, FOR ONE CHANNEL

TASK DEFINITION

```
TASK  004
title CHANGE

Function Scan
trigger Timer
delay 00-00:00:00.0
scans *
scan Inter
interval 00-00:00:02
.0
channels 001
rate 10/s
processing On
data to be logged Pr
oc
header LogData
output device Print
Format Full
tasks triggered 0
tasks aborted 0
```

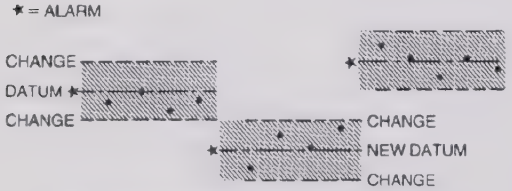
PRINT-OUT

```
1  +000 IF FLAG=0 THEN
   +001 FLAG=1
   +002 DATUM=C1
   +003 CHANGE=0.01
   +004 PRINT C1;1;"VDC
   "
   +005 ENDIF
2  +006 IF C1>DATUM+CHA
   NGE OR C1<DATUM-CHAN
   GE THEN
   +007 DATUM=C1
   +008 PRINT C1;1;"VDC
   "
   +009 ENDIF
```

PROGRAM

```
RUN    11:29:20 08-03
  CHANGE
S T 4 11:29:20.9
 :
P 001 1.33100  VDC .

D T 4
  CHANGE
S T 4 11:31:09.7
 :
P 001 1.34200  VDC .

D T 4
  CHANGE
S T 4 11:32:27.3
 :
P 001 1.40500  VDC .

D T 4
  CHANGE
S T 4 11:33:42.4
 :
P 001 1.39400  VDC .

D T 4
  CHANGE
S T 4 11:34:34.3
 :
P 001 0.998000 VDC .

D T 4
```

Change A value of change is entered.
The first result always constitutes an alarm, and the value of this result becomes the datum level for observing changes. Thereafter, an alarm occurs only when the result lies outside the datum ± change levels. When an alarm occurs, the datum moves to the value of the alarm result, and subsequent readings are compared with the new datum ± change. Each subsequent alarm causes the datum to adopt the new alarm value.



SUCCESSIVE RESULTS FROM ONE CHANNEL

PROGRAM KEY

1 This IF . . . THEN statement is actioned only on the first scan after RUN has been pressed, i.e. this is the only time that FLAG=0.
DATUM is set to the first reading on Channel 1.
The minimum change that will be recognised is defined as 0.01; the units being Volts in this case, as declared in the Channel Definition.
The initial value of C1 is output to the device specified in the Task Definition, in this case the Printer.

2 If a reading on C1 differs from the DATUM value by more than the permitted CHANGE amount, this reading becomes the new DATUM value, and is also output to the Printer.

15.7 OUTPUT ON CHANGE, GIVING OLD AND NEW VALUES, AND THE DIFFERENCE BETWEEN THEM

TASK DEFINITION

```
TASK  001
title CHANGE

Function Scan
trigger Timer
delay 00-00:00:00.0
scans *
scan Inter
interval 00-00:00:02
.0
channels 001
rate 10/s
processing On
data to be logged Pr
oc
header LogData
output device Print
format Full
tasks to be set 0
tasks aborted 0
```

PROGRAM

```
1000 IF FLAG=0 THEN
1001 FLAG=1                    1
1002 DATUM=C1                  1
1003 CHANGE=0.01               1
1004 PRINT DATUM;"NE           1
WDATUM";2;"VDC"                1
1005 ELSE                      2
1006 DIFF=ABS(C1-DAT           3
UM)
1007 IF DIFF>CHANGE            4
THEN
1008 OLDATUM=DATUM             5
1009 DATUM=C1                  6
1010 PRINT OLDATUM;"           7
OLDATUM";1;"VDC"               7
1011 PRINT DATUM;"NE           7
WDATUM";2;"VDC"                7
1012 PRINT DIFF;"DIF           7
FERENCE IS";3;"VDC"            7
1013 ENDIF
1014 ENDIF
```

PRINT-OUT

```
RUN    17:10:16 08-03
  CHANGE
S T 1 17:10:16.8
 :NEWDATUM
P 002 5.00100  VDC .

D T 1
  CHANGE
S T 1 17:14:16.0
 :O_DATUM
P 001 5.00100  VDC .
 :NEWDATUM
P 002 5.01200  VDC .
 :DIFFERENCE IS
P 003 0.0109998VDC .

D T 1
  CHANGE
S T 1 17:15:08.6
 :OLDATUM
P 001 5.01200  VDC .
 :NEWDATUM
P 002 4.98300  VDC .
 :DIFFERENCE IS
P 003 0.0290007VDC .

D T 1
  CHANGE
S T 1 17:15:48.2
 :O_DATUM
P 001 4.98300  VDC .
 :NEWDATUM
P 002 5.00400  VDC .
 :DIFFERENCE IS
P 003 0.0210008VDC .

D T 1
  CHANGE
S T 1 17:16:31.1
 :O_DATUM
P 001 5.00400  VDC .
 :NEWDATUM
P 002 5.01600  VDC .
 :DIFFERENCE IS
P 003 0.0119998VDC .

D T 1
```

PROGRAM KEY

See previous program for description of purpose.

1 As for previous program.

2 Program jumps straight to this line on all scans other than the first.

3 The difference (DIFF) is defined as the *absolute* value of the difference between the latest reading on Channel 1 and the previously stored reading in DATUM. Note that polarity is disregarded.

4 Only if the difference is greater than the permitted change, i.e. 0.01, will the rest of the program be executed. If the difference is less than CHANGE, the program is terminated for that scan, and nothing is printed.

5 The previous value in DATUM is transferred to OLDATUM.

6 The new reading on Channel 1 is stored in DATUM.

7 The three results, i.e. the old reading, the new reading and the difference between them, are output to the device specified in the Task Definition, in this case the Printer.

15.8 WARNING IF CHANNEL READING IS OUTSIDE LIMITS

TASK DEFINITION

```
TASK  001
title WARNINGS

Function Scan
trigger Timer
delay 00-00:00:00.0
scans *
scan Inter
interval 00-00:00:02
.0
channels 001
rate 10/s
processing On
data to be logged Pr
oc
header LogData
output device Print
Format Full
tasks triggered 0
tasks aborted 0
```

PROGRAM

```
1 1000 HIGH=5.2
2 1001 LOW=4.7
3 1002 IF C1>HIGH THEN
4 1003 PRINT> C1;"WARN
  ING HIGH LIMIT";1;"V
  DC"
  1004 ELSE
5 1005 IF C1<LOW THEN
6 1006 PRINT< C1;"WARN
  ING LOW LIMIT";1;"VD
  C"
  1007 ENDIF
  1008 ENDIF
```

PRINT-OUT

```
RUN    09:36:30 08-03
   WARNINGS
S T 1 09:36:30.9
  !WARNING HIGH LIMIT
P 001 5.21032  VDC>.

D T 1
   WARNINGS
S T 1 09:38:01.8
  !WARNING LOW LIMIT
P 001 4.69999  VDC<.

D T 1
```

PROGRAM KEY

1 High limit is defined.

2 Low limit is defined.

3 If the Channel reading is greater than the high limit:

4 Output the result, in alarm format.

5 If the Channel reading is less than the low limit:

6 Output the result, in alarm format.

15.9 ALARM THRESHOLD

TASK DEFINITION

```
TASK  001
title THRESHOLD

Function Scan
trigger Timer
delay 00-00:00:00.0
scans *
scan Inter
interval 00-00:00:02
.0
channels 001
rate 10/s
processing On
data to be logged Pr
oc
header LogData
output device Print
Format Full
tasks triggered 0
tasks aborted 0
```

PROGRAM

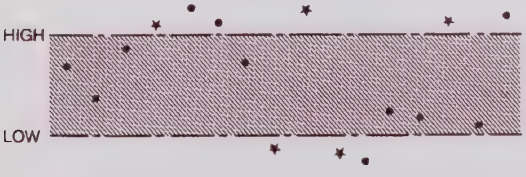
```
1  1000 IF START=0 THEN
   1001 START=1
   1002 HIGH=5.2
   1003 LOW=4.8
   1004 ENDIF
2  1005 IF C1>HIGH AND
   FLAG<>1 THEN
   1006 PRINT> C1;">HIG
   H THRESHOLD";1;"VDC"
3  1007 FLAG=1
   1008 ENDIF
4  1009 IF C1<LOW AND F
   LAG<>2 THEN
   1010 PRINT< C1;"<LOW
    THRESHOLD";1;"VDC"
5  1011 FLAG=2
   1012 ENDIF
```

PRINT-OUT

```
RUN    14:06:58 08-03
  THRESHOLD
S T 1 14:06:58.8
 !>HIGH THRESHOLD
P 001 5.21000  VDC>.

D T 1
  THRESHOLD
S T 1 14:08:10.2
 !<LOW THRESHOLD
P 001 4.79000  VDC<.

D T 1
  THRESHOLD
S T 1 14:09:21.2
 !>HIGH THRESHOLD
P 001 5.22300  VDC>.

D T 1
  THRESHOLD
S T 1 14:10:03.7
 !<LOW THRESHOLD
P 001 4.88900  VDC<.

D T 1
```

PROGRAM KEY

This program is based on a combination of the Alarm thresholds. A high limit and a low limit are defined the area between them constituting a "dead band". The first reading outside this band is output, but thereafter a reading will be output only when the entire band has been crossed, as shown below:



* = ALARM

SUCCESSIVE RESULTS FROM ONE CHANNEL

1 Actioned solely on the first scan, the only time when START=0. The high and low limits are defined.

2 If the reading on Channel 1 exceeds the high limit *and* FLAG does not equal 1, output the result in alarm format. Note that on the first scan, FLAG will be zero, therefore one of the conditions has been met, and the decision whether to output or not will depend solely on the reading.

3 If a high alarm has been registered, FLAG is assigned the value 1. This ensures that no further high alarms can be output until FLAG has been assigned a value other than 1, either by a low alarm or by stopping and re-starting scanning again via the RUN key.

4 If the reading on Channel 1 is less than the low limit *and* FLAG does not equal 2, output the result in alarm format. Note that on the first scan, FLAG will be zero, therefore the conditions are similar to those for line 1005 described above. However, the reading cannot be simultaneously *above* the high limit and *below* the low limit, so only one of the IF . . . THEN statements will be actioned.

5 If a low alarm has been registered, FLAG is assigned the value 2. This ensures that no further low alarms can be output until FLAG has been assigned a value other than 2, either by a high alarm or by stopping and re-starting scanning again via the RUN key.

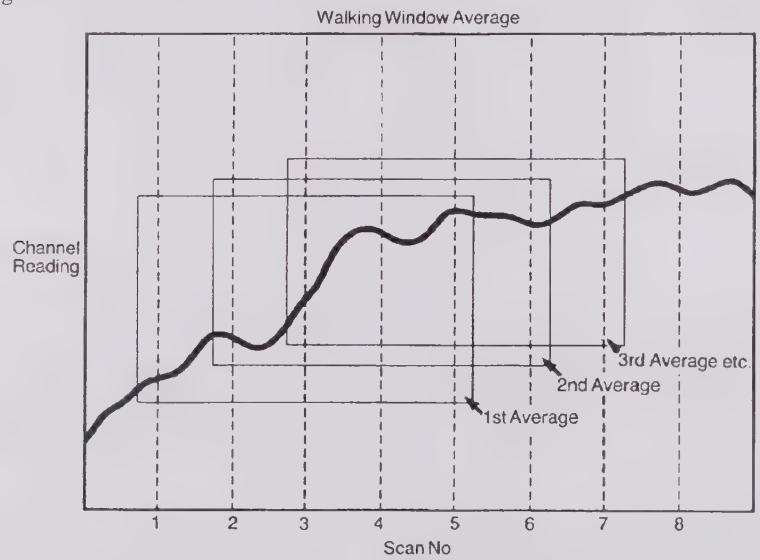
15.10 WALKING WINDOW AVERAGES

Previous examples show two methods of calculating averages:

1. Calculating a running average, with an output on every scan. When used for detecting trends, this method has the disadvantage that, once a large number of readings has been accumulated, each new reading has a barely perceptible effect on the result. Any trend becomes apparent only after several readings of either increasing or decreasing value have been added.

2. Calculating the average over groups of, say, 5 scans, with an output on every fifth scan. This method highlights trends much more clearly than the previous one, but has the disadvantage that a delay of 5 scans must elapse between consecutive readings. This could allow a trend to develop to dangerous extent before it is detected.

The walking window average offers a solution to these problems. An arbitrary number of readings is selected, say 5. Assume for the moment that the first 5 scans have been completed, the average calculated and the result output. On the 6th scan, the reading from scan 1 is discarded and the average of the readings from scans 2 to 6 is calculated, as shown in the figure below:



This process is repeated so that only the average of the latest 5 scans is output after each scan.

During the first scan, a special section of the program is used to ensure that there are always *five* readings to be averaged, even though only the first scan has been performed. In this case, the first reading is stored in the memory five times, to simulate five consecutive readings. The average is then calculated from these, thus yielding the result of just the first measurement.

After the second scan, the average is calculated from one scan 2 reading and four scan 1 readings. The table below shows how the "average" from the first 5 scans is calculated.

	NUMBER OF READINGS FROM SCAN:				
AFTER SCAN NUMBER:	1	2	3	4	5
1	5	–	–	–	–
2	4	1			
3	3	1	1	–	–
4	2	1	1	1	–
5	1	1	1	1	1

15.11 WALKING WINDOW AVERAGE ON ONE CHANNEL

TASK DEFINITION

```
TASK  001
title WALKING WINDOW

function Scan
trigger Timer
delay 00-00:00:00.0
scans *
scan Inter
interval 00-00:00:02
.0
channels 001
rate 10/s
processing On
data to be logged Al
l
header LogData
output device Print
Format Full
tasks triggered 0
tasks aborted 0
```

PROGRAM

```
1  1000 WINDOW:5
   1001 IF POINTER=0 TH
2  EN
   1002 FOR I=1 TO 5
   1003 WINDOW(I)=SC1
   1004 NEXT
   1005 ENDIF
3  1006 POINTER=POINTER
   +1
   1007 IF POINTER>5 TH
4  EN
   1008 POINTER=1
   1009 ENDIF
5  1010 WINDOW(POINTER)
   =SC1
   1011 AVERAGE=0
   1012 FOR I=1 TO 5
6  1013 AVERAGE=AVERAGE
   +WINDOW(I)
   1014 NEXT
7  1015 AVERAGE=AVERAGE
   /5
8  1016 PRINT AVERAGE;"
   AVERAGE";1;"VDC"
```

PRINT-OUT

C = Channel Reading
P = Processed Average Reading

```
RUN   14:50:35 13-03
  WALKING WINDOW
S T 1 14:50:35.7
 :
C 001 1.30007  Vdc
 :AVERAGE
P 001 1.30007  VDC .

D T 1
  WALKING WINDOW
S T 1 14:50:37.7
 :
C 001 1.40011  Vdc
 :AVERAGE
P 001 1.32008  VDC .

D T 1
  WALKING WINDOW
S T 1 14:50:39.7
 :
C 001 1.50025  Vdc
 :AVERAGE
P 001 1.36011  VDC .

D T 1
  WALKING WINDOW
S T 1 14:50:41.7
 :
C 001 1.40010  Vdc
 :AVERAGE
P 001 1.38012  VDC .

D T 1
  WALKING WINDOW
S T 1 14:50:43.7
 :
C 001 1.30007  Vdc
 :AVERAGE
P 001 1.38012  VDC .

D T 1
  WALKING WINDOW
S T 1 14:50:45.7
 :
C 001 1.20002  Vdc
 :AVERAGE
P 001 1.36011  VDC .

D T 1
  WALKING WINDOW
S T 1 14:50:47.7
 :
C 001 1.09989  Vdc
 :AVERAGE
P 001 1.30007  VDC .

D T 1

HALT  14:51:02 13-03
```

PROGRAM KEY

1 Declaration of 5-element variable, WINDOW(1) to WINDOW(5).

2 Used on first scan only, as described in previous section, to ensure that all elements contain a reading.

3 "Points out" which element is to store the reading.

4 Resets the Pointer after every fifth scan.

5 As the index "POINTER" is incremented from 1 to 5, so WINDOW becomes WINDOW(1), WINDOW(2) etc.; and successive scans cause values of Scanned Channel 1 to be stored in the correct "WINDOW".

6 At the end of each scan, the new sum of WINDOWS(1) to (5) will have been stored in AVERAGE.

7 AVERAGE now contains the average value of the last 5 readings.

8 Outputs the average value to the device specified in the Task Definition, in this case the Printer.

Note that the Task Definition specifies that "All" data is to be logged, hence each scan produces a print-out from the Scanned Channel, in this case Channel 1, and a processed result showing the average value.

15.12 WALKING WINDOW AVERAGE ON THREE CHANNELS, OVER FIVE READINGS

TASK DEFINITION

```
TASK  001
title WINDOW (3-CHAN
NEL)
Function Scan
trigger Timer
delay 00-00:00:00.0
scans *
scan Inter
interval 00-00:00:02
.0
channels 003,007,012

rate 10/s
processing On
data to be logged Al
l
header LogData
output device Print
Format Full
tasks triggered 0
tasks aborted 0
```

PROGRAM

```
1  1000 WINDOW;15
   1001 IF POINTER=0 TH
   EN
   1002 FOR N=1 TO 3
   1003 FOR I=1 TO 5
2  1004 WINDOW((N-1)*5+
   I)=SC(N)
   1005 NEXT
   1006 NEXT
   1007 ENDIF
3  1008 POINTER=POINTER
   +1
   1009 IF POINTER>5 TH
4  EN
   1010 POINTER=1
   1011 ENDIF
   1012 FOR N=1 TO 3
   1013 SUM=0
5  1014 WINDOW((N-1)*5+
   POINTER)=SC(N)
   1015 FOR I=1 TO 5
6  1016 SUM=SUM+WINDOW(
   (N-1)*5+I)
   1017 NEXT
7  1018 SC(N)=SUM/5
   1019 NEXT
```

PRINT OUT

```
RUN    16:33:59 13-03
  WINDOW (3-CHANNEL)
S T 1  16:33:59.9
  :
C 002 0.310451 Vdc .
  :
C 006 0.323107 Vdc .
  :
C 014 0.317229 Vdc .

D T 1
  WINDOW (3-CHANNEL)
S T 1  16:34:01.9
  :
C 002 0.306507 Vdc .
  :
C 006 0.316905 Vdc .
  :
C 014 0.312652 Vdc .

D T 1
  WINDOW (3-CHANNEL)
S T 1  16:34:03.9
  :
C 002 0.307233 Vdc .
  :
C 006 0.315985 Vdc .
  :
C 014 0.313516 Vdc .

D T 1

ETC
```

This program is similar to the previous one, but shows how to create a "two-dimensional" array so that readings from more than one Channel can be logged. The variable, WINDOW, has 15 elements, arranged in 3 rows of 5 each, as shown in the figure:

Scanned Channel	WINDOW				
First (e.g. C2)	1	2	3	4	5
Second (e.g. C6)	6	7	8	9	10
Third (e.g. C14)	11	12	13	14	15

PROGRAM KEY

1 Declaration of 15-element variable, WINDOW(1) to WINDOW(15).

2 Used on first scan only. The two nested FOR . . . NEXT loops ensure that all 15 elements contain a reading, for reasons explained in Section 15.10. Line 1004 behaves as follows:

When N=1, (N−1)*5=0.
Therefore, as I is incremented from 1 to 5, the reading from Scanned Channel 1 is stored in WINDOW(1) to WINDOW(5).

When N=2, (N−1)*5=5. Therefore, as I is incremented from 1 to 5, the reading from Scanned Channel 2 is stored in WINDOW(6) to WINDOW(10); and so on.

3 Points to the relevant column in the array, on each scan.

4 Resets POINTER after every fifth scan.

5 This line behaves in a similar manner to line 1004, except that POINTER is now used as part of the index for WINDOW. For example, on the third scan of a group of five, POINTER will equal 3. Hence, when N=1, WINDOW(3) will store the reading from Scanned Channel 1. When N=2, WINDOW(8) will store the reading from Scanned Channel 2, and when N=3, WINDOW(13) will store the reading from Scanned Channel 3.

6 This FOR . . . NEXT loop is nested within another loop which starts on line 1012. When the outer loop sets N to, say 2; the inner loop will cause the values in WINDOW(6) to WINDOW(10) to be added together and stored in SUM.

7 When n=2, for example, the latest reading stored from Scanned Channel 2, which is now no longer required, is replaced by the average value of the last 5 readings on Scanned Channel 2.

Note that no Print Statement is used; the three averages stored in SC(1), (2) and (3) are output as defined by the Task Definition. Hence, they appear as "C" results, although accompanied by the dot which denotes a processed result.

15.13 MEAN READING AND STANDARD DEVIATION OF 20 CHANNELS, OVER 10 SCANS

TASK DEFINITION

```
TASK  001
title STD.DEV.AND ME
AN
Function Scan
trigger Timer
delay 00-00:00:00.0
scans 10
scan Inter
interval 00-00:00:02
.0
channels 001-020
rate 40/s
processing On
data to be loaded Pr
oc
header LogData
output device Print
Format Full
tasks triggered 0
tasks aborted 0
```

PROGRAM

```
1  1000 MEAN;20;"MEAN";
   1;"VDC"
2  1001 DEV;20;"SD";1
3  1002 N=N+1
   1003 FOR I=1 TO 20
4  1004 MEAN(I)=MEAN(I)
   +C(I)
5  1005 DEV(I)=DEV(I)+C
   (I)**2
   1006 NEXT
6  1007 IF N=10 THEN
7  1008 FOR I=1 TO 20
   1009 DEV(I)=MEAN(I)*
8  MEAN(I)/N-DEV(I)
   1010 DEV(I)=SQR(ABS(
   DEV(I)/N))
9  1011 MEAN(I)=MEAN(I)
   /N
10 1012 PRINT MEAN(I),D
   EV(I)
11 1013 MEAN(I)=0
   1014 DEV(I)=0
   1015 NEXT
12 1016 N=0
   1017 ENDIF
```

PRINT OUT

FIRST 10 CHANNELS

```
RUN    00:39:24 01-Fe
  STD.DEV.AND MEAN
STRT 1 00:39:42.1
 :MEAN
P 001 1.65300   VDC .
 :SD
P 001 1.069E-3      .
 :MEAN
P 002 2.84200   VDC .
 :SD
P 002 8.734E-4      .
 :MEAN
P 003 0.918000  VDC .
 :SD
P 003 3.088E-4      .
 :MEAN
P 004 1.04400   VDC .
 :SD
P 004 5.348E-4      .
 :MEAN
P 005 3.55100   VDC .
 :SD
P 005 0             .
 :MEAN
P 006 2.21300   VDC .
 :SD
P 006 1.069E-3      .
 :MEAN
P 007 0.774000  VDC .
 :SD
P 007 0             .
 :MEAN
P 008 1.14400   VDC .
 :SD
P 008 6.176E-4      .
 :MEAN
P 009 1.97300   VDC .
 :SD
P 009 0             .
 :MEAN
P 010 2.06700   VDC .
 :SD
P 010 1.381E-3      .
```

PROGRAM KEY

1 Declares a variable, MEAN, having 20 elements (one per Channel).

2 Declares a variable, DEV, also having 20 elements, with normal text SD (i.e. Standard Deviation).

3 Tally Counter.

4 The readings from C1 to C20 are stored as variables MEAN(1) to MEAN(20).

5 The *squares* of the readings from C1 to C20 are stored as variables DEV(1) to DEV(20).

6 For the first 9 scans, N does not equal 10 and therefore the program jumps to line 1017 and terminates. On the 10th scan, the results are to be output and therefore the rest of the program is actioned.

7 Start of a FOR . . . NEXT loop, to process the stored results from all 20 Channels.

8 These two lines constitute an algorithm which re-arranges the text-book equation for Standard Deviation:

$$s = \sqrt{\frac{\sum_{i=1}^{n} (x_i - \bar{x})^2}{n}}$$

into a machine-compatible form.

Analysis of line 1009 reveals that DEV(I) will contain

$$\frac{(\Sigma x_i)^2 - \Sigma x_i^2}{N}$$

and line 1010 modifies this to give the square root of the absolute value, which, when divided by N, represents the Standard Deviation.

9 The mean value of the readings from each Channel is stored in MEAN(1) to MEAN(20).

10 Print Statement.

11 Set variables to zero after print-out

12 Set tally counter to zero after 10th scan.

Chapter 5
Output Format

Section		*Page*
1	**General Information**	5.3
1.1	Adding Text	5.3
1.2	Monitor Facility	5.5
2	**Output Data Formats**	5.5
2.1	Numerical Result	5.6
2.2	Start and End of Scan Markers	5.7
2.3	Event Result	5.7
2.4	Run and Halt Markers	5.8
2.5	Power Fail and Power Back Messages	5.9
2.6	Date	5.9
2.7	Transient Capture Data Output	5.10
2.7.1	Full Format	5.10
2.7.2	Raw Data Format	5.11
2.7.3	Binary and Coded Format	5.11
2.8	Transient Start Date Marker	5.12
2.9	Transient Start Time Marker	5.12
2.10	Transient Trigger Delay Time Marker	5.12
2.11	Transient Interval Time Marker	5.12
2.12	Transient Channel Result Marker	5.12
2.13	Transient Channel Maximum Result Marker	5.13
2.14	Transient Channel Minimum Result Marker	5.13
2.15	Number of Results in Capture	5.13
2.16	End of Export Marker	5.13
2.17	Export Abort Marker	5.13
2.18	Specific Formatting	5.14
3	**Coded Binary Formatting**	5.15
3.1	Numerical Result	5.17
3.2	Start and End of Scan Markers	5.21
3.3	Event Result	5.21
3.4	Run and Halt Markers	5.21
3.5	Power Fail and Power Back	5.22
3.6	Date	5.22
3.7	Transient Start Date Marker	5.22
3.8	Transient Start Time Marker	5.22
3.9	Transient Trigger Delay Time Marker	5.23
3.10	Transient Interval Time Marker	5.23
3.11	Number of Results in Capture	5.23
3.12	End of Export Marker	5.23
3.13	Transient Channel Maximum Result Marker	5.24
3.14	Transient Channel Minimum Result Marker	5.24
3.15	Export Abort Marker	5.24
3.16	Transient Channel Result	5.24

Section *Page*

4	**Binary Format**	5.25
4.1	Untagged Numerical Result	5.26
4.2	Time Tagged Numerical Result	5.27
4.3	Start of Scan Markers	5.27
4.4	End of Scan Markers	5.28
4.5	Event Result	5.28
4.6	Run Marker	5.28
4.7	Halt Marker	5.28
4.8	Power Fail	5.29
4.9	Power Back	5.29
4.10	Date	5.29
4.11	Transient Start Date	5.30
4.12	Transient Trigger Delay Time	5.30
4.13	Transient Interval Time	5.31
4.14	End of Export Marker	5.31
4.15	Transient Channel Result	5.31
4.16	Transient Capture Start Time	5.31
4.17	Export Abort Marker	5.32
4.18	Maximum Result in Capture	5.32
4.19	Number of samples in Capture	5.32
4.20	Minimum Result in Capture	5.32
5	**Data Blocking**	5.33
5.1	Data Blocking Commands	5.33
5.2	Data Blocking Output Conditions	5.34
6	**Status Variables**	5.34
7	**Storing Data on Disk (SI3535 D & G)**	5.36
7.1	Normal/Cyclic Storage Modes	5.36
7.2	Disk Formats	5.36
7.3	Data Output Format	5.37
7.4	SCORPIO Disk Program	5.37
7.5	Disk Compatibility	5.37
7.6	How to replay the disk in another machine	5.38

Tables *Page*

5.1	Event Text Output	5.4
5.2	Status Query Information	5.35

1 GENERAL INFORMATION

The user may select the format in which data, ie messages, measured results, markers etc., will be output by SCORPIO. Output formats are available: Full, Compact, Coded, Binary format, Raw Data and Full Printing Format.

Full: 40 character ASCII string, contains full information including text messages defined by the text command. Primarily designed for display at a VDU or printer.

Compact: 20 character ASCII string, contains full numeric data (results, channel nos. etc.) and SCORPIO generated messages, but does not include text defined by the text command.

Coded: 10 character ASCII string, contains numeric data in binary form, encoded as printable ASCII characters. No text of any type is included. This format is intended primarily for storing data on a SI3535 D or G disk or another data collection device connected to one of the interfaces.

Note: When processed data is stored on a SI3535D or G disk each processed data record may be followed by a further 10 character record containing information which will allow the SCORPIO to reconstitute the processed data if replayed from disk. Further details of binary coded processed data are given later in this chapter.

Binary: 8 or 14 byte string, containing data in non-printable binary format. Data cannot be output or replayed from the logger disk. This format is used to optimise data throughput to a computer or other controller via RS232 or GPIB interfaces

Transient Capture :

Full : Data in the full format can be read by the operator and is used in place of the full and compact formats reserved for scan results. Data is output in 20-character lines and consists of six header lines, the results and three footer lines.

Raw Data: Raw data format is used to export transient results as quickly as possible. In this format the 16-bit unlinearized data from the transient capture ADC is exported, together with a header that tells a computer how to process the data.

Coded: As above.

Binary: As above.

When the output format is not specified the SCORPIO will default to output in Full format.

1.1 ADDING TEXT

Any output result can be accompanied by a user-defined text comprising a short message. Text is only output in Full Format, no text is output in Compact or Coded Format (Format is selected in the Task Definition). There are two types of text: non-alarm text and alarm text. Upto 600 non-alarm and alarm-only texts can be entered. One non-alarm text and/or one alarm text can be assigned to any input channel, or group of channels.

Each text will then be automatically output with a result as follows:

Non-alarm text - output with any non-alarm result
Alarm text - output with any alarm result

Each text, which has an exclusive number in the range 1-600, is attached to a specific input channel (or group of channels) by entering the appropriate text number in the Text part of the Channel Definition. Any defined text can be used as a non-alarm or alarm-only text, and any text can be used in any number of Channel Definitions.

A simple example is shown below.

Text Definition:	Text No. 1 = MESSAGE A Text No. 2 = MESSAGE B
Channel Definition for Channel n:	Non-alarm text number = 1 Alarm text number = 2

MESSAGE A and MESSAGE B are written and entered using the Text Definition, and the text numbers of these messages are entered in the Channel Definition for channel n. This will result in the following automatic action:

All non-alarm results from channel n will be accompanied by MESSAGE A

All alarm-only results from channel n will be accompanied by MESSAGE B

Channel n could, for instance, be a thermocouple channel used to measure temperatures on a boiler. MESSAGE A could state 'BOILER O.K.', whereas MESSAGE B, output when the temperature had exceeded an alarm level, could state 'BOILER OVERHEATING'.

The text that accompanies a processed result is dependent upon the user program. Results from Event inputs will be accompanied by text as follows:

Event, hi-to-lo transition - non-alarm text

Event, lo-to-hi transition - alarm-only text

All conditions for text output are summarised in Table 5.1.

Table 5.1 *Event Text Output*

TEXT TYPE	RESULT TYPE			
	NON-ALARM LOG	ALARM LOG	EVENT hi to lo	EVENT lo to hi
Non-alarm	●		●	
Alarm only		●		●

● = text output

1.2 MONITOR FACILITY

Full or Compact format may be selected for monitoring, using the Monitor facility (MON). Note that the format selected for monitoring operates independently of the format for logged data, selected using a Task Definition.

Monitored results are distinguished from normal logged results as follows:

Monitored results are preceded by M
e.g. M 002 1.6554 V

Logged results are preceded by a C
e.g. C 002 1.6554 V

2 OUTPUT DATA FORMATS

The data output by SCORPIO is dependent on logger action and can be grouped into types as follows:

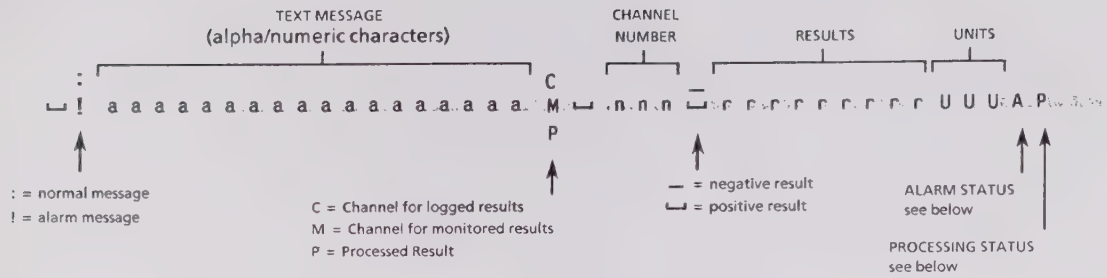
1) Numerical results
2) Start and end scan markers
3) Event results
4) Run and Halt markers
5) Power Fail and Power Back messages
6) Date
7) Transient Start Date Marker
8) Transient Start Time Marker
9) Transient Trigger Delay Time Marker
10) Transient Interval Time Marker
11) Transient Channel Result
12) Transient Channel Maximum Result Marker
13) Transient Channel Minimum Result Marker
14) Number of Results in Capture Marker
15) Transient End of Export Marker
16) Transient Export Abort Marker

The first character of each data string identifies the output format of that string.

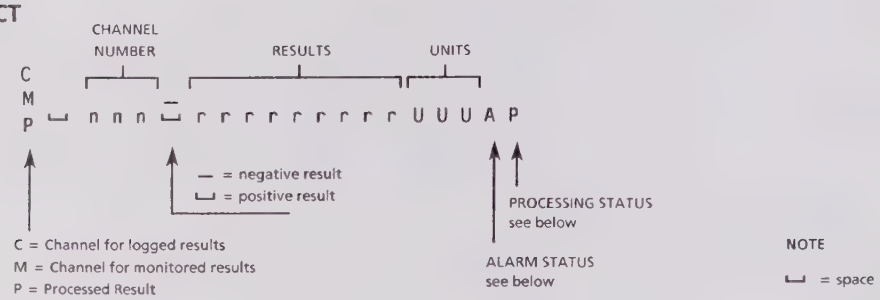
This section gives details of the standard layout of each of the data types. Section 3 describes how the binary data is encoded to produce a Coded format record and section 4 describes the binary data ouput format.

2.1 NUMERICAL RESULT

FULL



COMPACT



ALARM STATUS

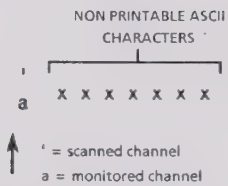
- `<` = result is less than low alarm limit (Limit, Threshold or Change)
- `>` = result is greater than high alarm limit (Limit, Threshold or Change)
- `!` = Pattern Match or Mismatch, or first Change result (subsequent Change Results show < or Z)
- ␣ = non-alarm reading

PROCESSING STATUS

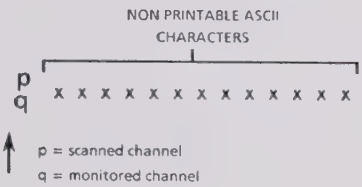
- ␣ = non-processed result
- • = processed result

BINARY

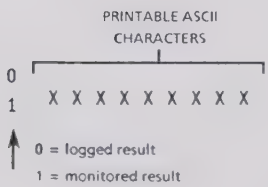
(Not time tagged)



(Time tagged)

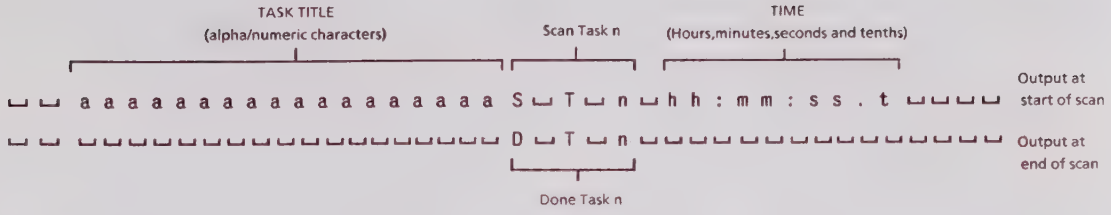


CODED

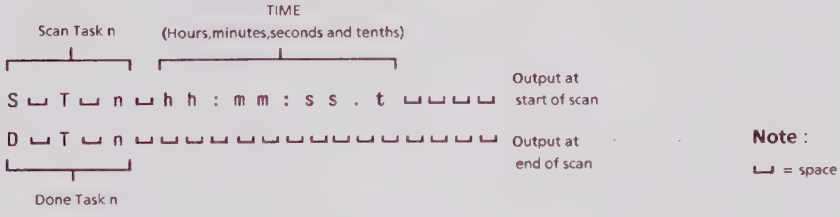


2.2 START AND END OF SCAN MARKERS

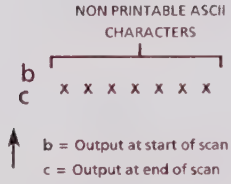
FULL


TASK TITLE
(alpha/numeric characters)
Scan Task n
TIME
(Hours,minutes,seconds and tenths)
␣ ␣ a a a a a a a a a a a a a a a a a a S ␣ T ␣ n ␣ h h : m m : s s . t ␣ ␣ ␣ ␣
Output at start of scan
␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ D ␣ T ␣ n ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣
Output at end of scan
Done Task n

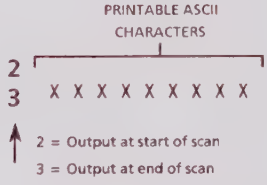
COMPACT


Scan Task n
TIME
(Hours,minutes,seconds and tenths)
S ␣ T ␣ n ␣ h h : m m : s s . t ␣ ␣ ␣ ␣
Output at start of scan
D ␣ T ␣ n ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣ ␣
Output at end of scan
Done Task n
Note :
␣ = space

BINARY

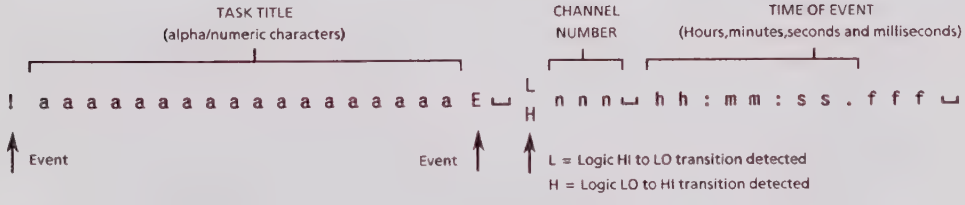


CODED

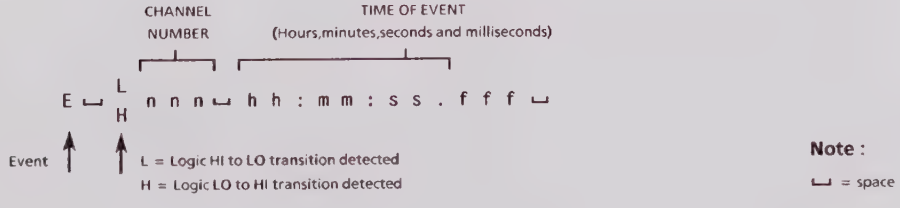


2.3 EVENT RESULT

FULL

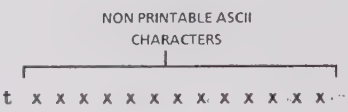

TASK TITLE
(alpha/numeric characters)
CHANNEL NUMBER
TIME OF EVENT
(Hours,minutes,seconds and milliseconds)
! a a a a a a a a a a a a a a a a a a E ␣ L H n n n ␣ h h : m m : s s . f f f ␣
Event
Event
L = Logic HI to LO transition detected
H = Logic LO to HI transition detected

COMPACT

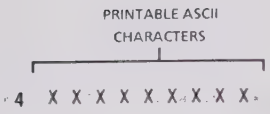

CHANNEL NUMBER
TIME OF EVENT
(Hours,minutes,seconds and milliseconds)
E ␣ L H n n n ␣ h h : m m : s s . f f f ␣
Event
L = Logic HI to LO transition detected
H = Logic LO to HI transition detected
Note :
␣ = space

EVENT RESULT [continued]

BINARY



CODED

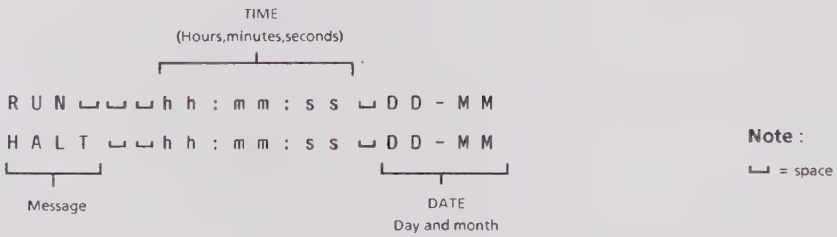


2.4 RUN AND HALT MARKERS

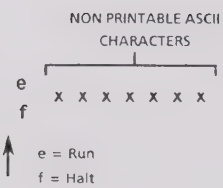
FULL

TASK TITLE
(alpha/numeric characters)
TIME
(Hours,minutes,seconds)
␣ ␣ a a a a a a a a a a a a a a a a a a R U N ␣ ␣ ␣ h h : m m : s s ␣ D D - M M
␣ H A L T ␣ ␣ h h : m m : s s ␣ D D - M M
Message
DATE
Day and month

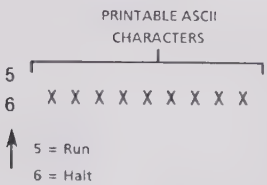
COMPACT



BINARY

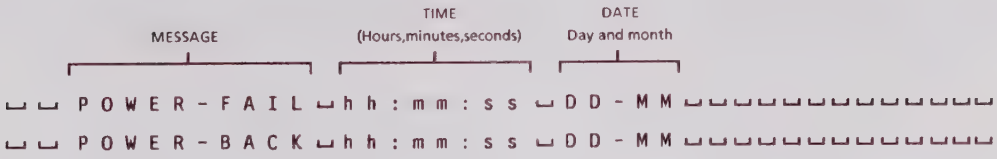


CODED

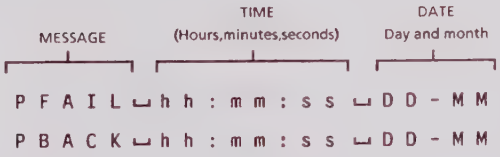


2.5 POWER FAIL AND POWER BACK MESSAGES

FULL

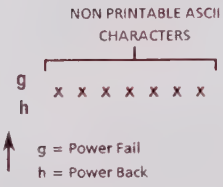


COMPACT

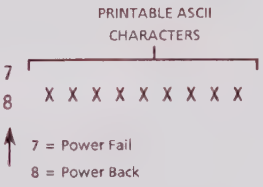


Note :
␣ = space

BINARY



CODED



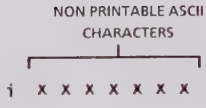
2.6 DATE

FULL

COMPACT

Note :
␣ = space

BINARY



CODED

2.7 TRANSIENT CAPTURE DATA OUTPUT

The selection of output formats and output devices applicable in transient capture, are shown below:

OUTPUT DEVICE	FORMATS AVAILABLE			
	Full	Coded	Binary	Raw
Display	•			
Printer	•			
Disk	•	•		
GPIB	•	•	•	•
RS232	•	•	•	•
RS422a,b	•	•		

• = format available.

Due to the fact that a captured transient consists of many results, the output formats available for transient capture data are somewhat different to the output formats available for scan results. The characteristics of each transient capture format are summarized as follows.

2.7.1 FULL FORMAT

Data in the full format can be read by the operator and is used in place of the full and compact formats reserved for scan results. Data is output in 20-character lines and consists of six header lines, the results and three footer lines, as follows:

T␣T␣a a a a a a a a a a a a a a a a	Transient text label.
T␣D␣c c c␣␣␣␣␣d d - M M - y y	Transient start date.
T␣S␣␣␣h h : m m : s s . u u u u u	Transient start time.
T␣O z h h h h : m m : s s . u u u u u	Transient trigger delay time.
T␣I␣␣h h h : m m : s s . u u u u u	Transient interval time.
T␣N␣␣n n n n n n␣␣␣␣␣␣␣␣␣	Number of results captured.
⋮	⋮
T␣c c c z r r r r r r r r U U U t ␣	Transient channel results
⋮	⋮
T␣H␣␣z r r r r r r r r U U U ␣␣	Transient channel maximum result.
T␣L␣␣z r r r r r r r r U U U ␣␣	Transient channel minimum result.
T␣A␣c c c␣␣␣␣␣␣␣␣␣␣␣␣␣␣	Transient end of export or
T␣E␣c c c␣␣␣␣␣␣␣␣␣␣␣␣␣␣	Transient export aborted.

2.7.2 RAW DATA FORMAT

Raw data format is used to export transient results as quickly as possible. In this format the 16-bit unlinearized data from the transient capture ADC is exported, together with a header that tells a computer how to process the data. The data that is output in raw format is summarized below.

Although the data is in different fields, no terminator characters are inserted between the start of the data block and the end.

The content of a block of raw data is as follows:

.	ASCII character "." , introducer.
nnnn	Number of bytes in transfer (4-byte integer).
aaaaaaaaaaaaaaaa	Transient text label (16 ASCII characters).
ccc	Channel number, 1 through 200 (ASCII).
dd-mm-yy	Start date (ASCII).
hh:mm:ss:uuuuu	Start time (ASCII).
shhhh:mm:ss:uuuuu	Trigger delay time (ASCII).
hhh:mm:ss:uuuuu	Interval time (ASCII).
NNNN	Number of results (4-byte unsigned integer).
XXXX	Result multiplier (4-byte floating point number).
OOOO	Result offset (4-byte integer).
R1	⋮
R2	16-bit results (two bytes; twos complement).
:	⋮
:	
RNNNN	
8000H	End of block normal.
8001H	End of block aborted.

All data in a raw data block is "little endian", i.e. the numbers are output with the least significant byte first.

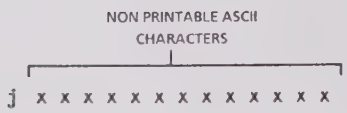
2.7.3 BINARY AND CODED FORMAT

Binary Format
Binary Format is the most convenient way of exporting data (via the GPIB or the RS 232 or RS 422a,b serial interface) for subsequent processing by computer. (Binary format needs no complex decoding, prior to being used by the computer.)

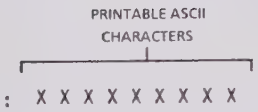
Coded Format
Coded Format is intended primarily for recording data on the logger disk, but data thus recorded may be replayed to any other device. Each result in coded format is a binary number encoded into a string of ten printable ASCII characters (includes <DELETE>). Texts are not output in coded form, so the transient text label is not included with the captured results: however, all the numeric information listed above for the full format is included. Coded format takes up minimum disk space and therefore permits the maximum number of results to be recorded. Results recorded in coded format are normally replayed only in the logger.

2.8 TRANSIENT START DATE MARKER

BINARY

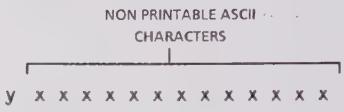


CODED

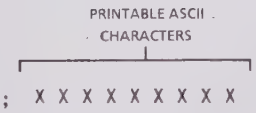


2.9 TRANSIENT START TIME MARKER

BINARY



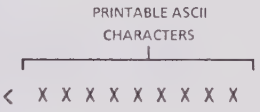
CODED



2.10 TRANSIENT TRIGGER DELAY TIME MARKER

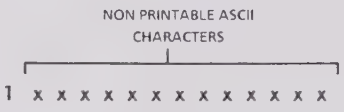
BINARY

CODED

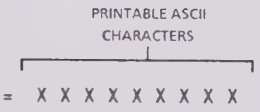


2.11 TRANSIENT INTERVAL TIME MARKER

BINARY

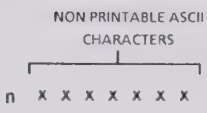


CODED

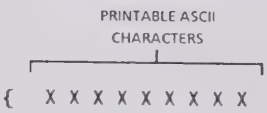


2.12 TRANSIENT CHANNEL RESULT

BINARY

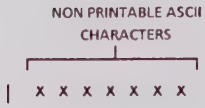


CODED

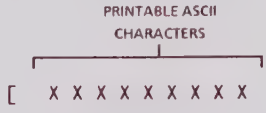


2.13 TRANSIENT CHANNEL MAXIMUM RESULT MARKER

BINARY

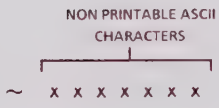


CODED

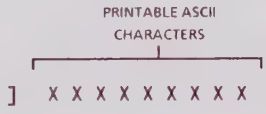


2.14 TRANSIENT CHANNEL MINIMUM RESULT MARKER

BINARY

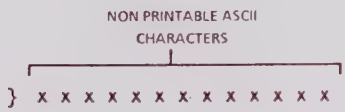


CODED

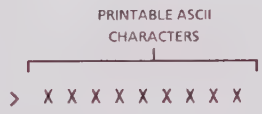


2.15 NUMBER OF RESULTS IN CAPTURE

BINARY

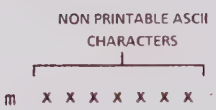


CODED



2.16 END OF EXPORT MARKER

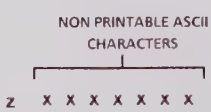
BINARY



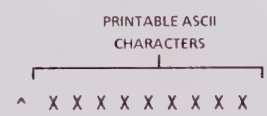
CODED

2.17 EXPORT ABORT MARKER

BINARY



CODED



2.18 SPECIFIC FORMATTING

In addition to the general format information given on the preceding pages, the following formatting details apply to the specified output devices only.

Disk : Data is compatible with IBM MSDOS (version 3.2) or HP-LIF formats.

Printer Terminal connected to PL1 on Controller (RS232) : Data is output in lines of up to 80 characters, terminated by CR or CR/LF and 4 nulls as determined by SW2 on Controller. Measurements are packed 2, 4 or 8 per line depending on Format. (The VDU mode supresses LF if line contains exactly 80 characters).

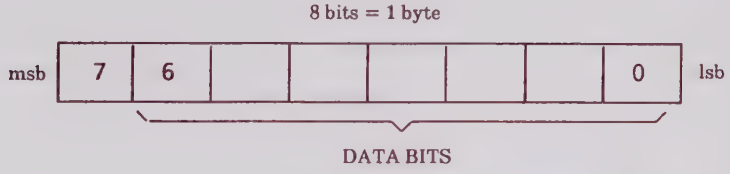
GP-IB devices, talker only mode : Data is output in lines of up to 80 characters, terminated by CR/LF. Measurements are packed 2, 4 or 8 per line depending on Format. (See Chapter 7 for details.)

RS422 ports a and b (Serial Interface) : Data is output in lines of up to 80 characters, terminated by CR, or CR/LF and 4 nulls. Measurements are packed 2, 4 or 8 per line depending on Format.(see Chapter 6 for details.)

The terminating characters (e.g. CR, CR/LF etc.) are additional to the data characters.

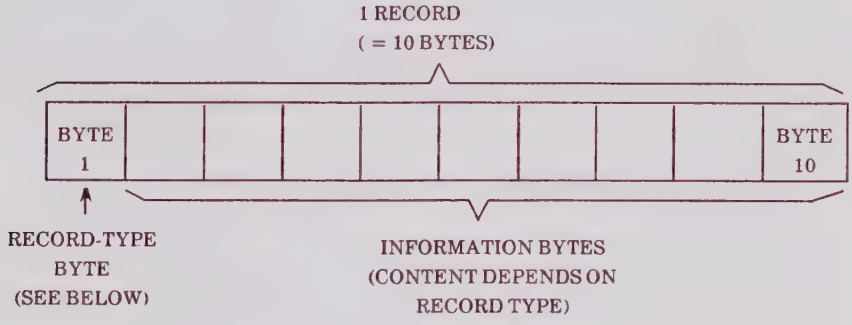
3 CODED FORMAT

Records output in Coded format consist of 10 byte packets. Each byte, which is encoded as a printing ASCII character, consists of:



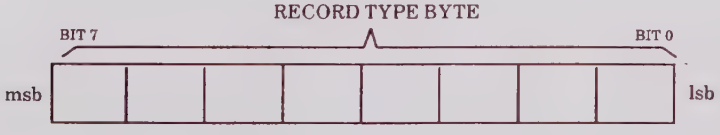
The most significant bits (bit 6 and 7) are normally output as '0' and '1', respectively. This makes the byte into a printing ASCII character.

All records consist of 10 bytes, although certain processed results consist of two records. The content varies according to the record type. The type, which depends on logger action at any given time, is indicated by the first byte in each record.



Record -Type Byte

The various types of Coded Format records are listed below.



[Ascii]	*[Hex]*	*Type*
0	30	Numerical Result: normal logged result
1	31	Numerical Result: monitored result
2	32	Start of Scan Marker
3	33	End of Scan Marker
4	34	Event Result
5	35	Run Marker
6	36	Halt Marker
7	37	Power Back
8	38	Power Fail
9	39	Date

[Ascii]	*[Hex]*	*Type*
:	3A	Transient Start Date Marker
;	3B	Transient Start Time Marker
<	3C	Transient Trigger Time Marker
=	3D	Transient Interval Time Marker
>	3E	Number of Results in Capture Marker
?	3F	Transient End of Export Marker
[	5B	Transient Channel Maximum Result Marker
]	5D	Transient Channel Minimum Result Marker
^	5E	Transient Export Abort Marker
{	7B	Transient Channel Result Marker

All Record types are detailed in the pages that follow. Each Record is shown in diagramatic form. The diagrams illustrate the overall content of the Records: the upper shows how the data is incorporated in the 10 byte Record which is output by the logger, the lowest part of each diagram shows the numeric data which is generated within the logger.

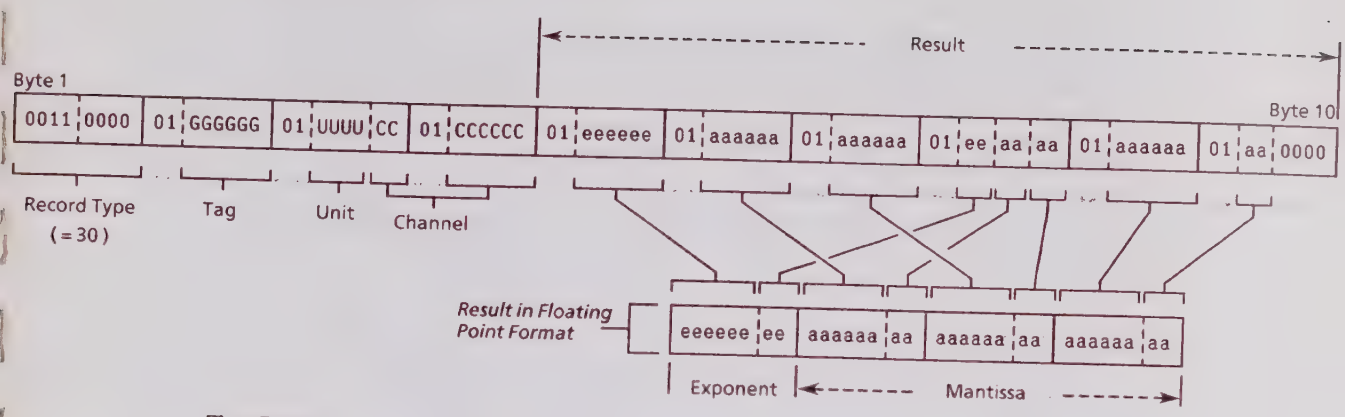
The coded data record contains all the numeric information that the equivalent full and compact format records contain. However, for certain record types, the coded record contains additional information not found in the full or compact record, ie:

a) The event record contains the Task number.
b) The Halt, Run, Power fail, Power Back and Date records have the year included in the date.

The following symbols are used in the format diagrams :

Symbol	Meaning
G	Tag
T	Task number
C	Channel number
U	Unit number
R	Number of results in capture
a	IEEE mantissa
e	IEEE exponent
h	Hours
m	Minutes
s	Seconds
t	Tenths
l	Milliseconds
c	Tens of Microseconds
Y	Year
M	Month
D	Day

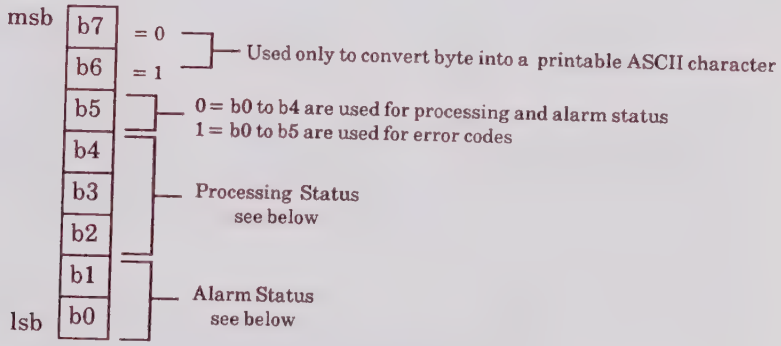
3.1 NUMERICAL RESULT



Tag Bytes

These are included in Numerical and Event results. A tag byte can convey processing and alarm status, or error codes

The tag byte is eight bits wide, although the two most significant bits are used only to make the byte into a printable ASCII character. The significance of each bit is described as follows :-



Processing Status

b4	b3	b2	
0	0	0	No processing
0	0	1	Processed result, where original Channel reading is overwritten
1	0	1	Derived processed result, where original Channel reading still exists

Alarm Status

b1	b0	*All measurement except Event detection*	*Event detection Only*	*Transient Capture*
0	0	Non-alarm result	Invalid condition	No tag byte
0	1	< Low limit	Logic Hi-to-Lo transition	Pre-trigger result
1	0	> High limit	Logic Lo-to-Hi transition	Post-trigger result
1	1	Invalid measurement condition or Match / Mismatch detected	Invalid measurement condition	Trigger result

Error Codes

Codes						Error Numbers	
b5	b4	b3	b2	b1	b0		
1	1	1	1	1	1	1	Measurement Overload
1	1	1	1	1	0	2	Measurement out of linearisation range
1	1	1	1	0	1	3	Invalid ambient temperature
1	1	1	1	0	0	4	Event total exceeds 65536
1	1	1	0	1	1	5	Timeout in external-module measurement (>20 secs)
1	1	1	0	1	0	6	Strain gauge channel not initialised
1	1	1	0	0	1	7	Input connector not fitted
1	1	1	0	0	0	8	Result not yet accumulated (Counter Timer module)
1	1	0	1	1	1	9	Counter Timer Channel is off

The error codes represent the 2's complement of the error number. If required, the error code may be extended by 2 bits to become an 8 bit 2's complement number, using external encoding. In this case bits 6 and 7 become '1', and the 8 bit number is read as 2's complement to obtain the error number.

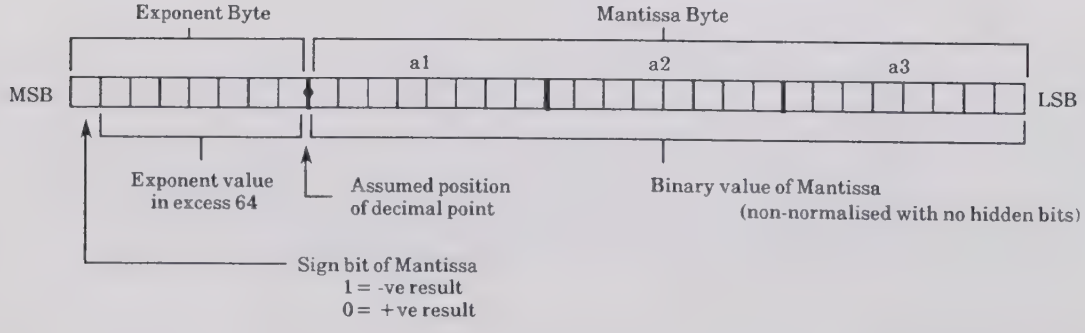
UNIT AND CHANNEL NUMBER BYTES

These produce a Composite Channel Number which indicates Unit (Master. 1st Slave, or 2nd Slave) and number within that unit (which is always in the range 1 to 200). However, in the case of the second record of a derived processed result, the Channel number is zero, indicating that the remainder of the record has no validity outside the logger.

Examples: Channel 005 (ie. in the Master Unit) is shown as 1011 00000101
Channel 205 (ie. in the 1st Slave Unit) is shown as 1100 00000101
Channel 405 (ie. in the 2nd Slave Unit) is shown as 1101 00000101

EXPONENT AND MANTISSA

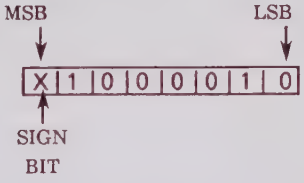
Each result is exprressed in standard floating point format, as an exponent and mantissa (a1, a2 and a3). The sign bit for the mantissa is contained in the Exponent byte (msb).



Exponent Byte

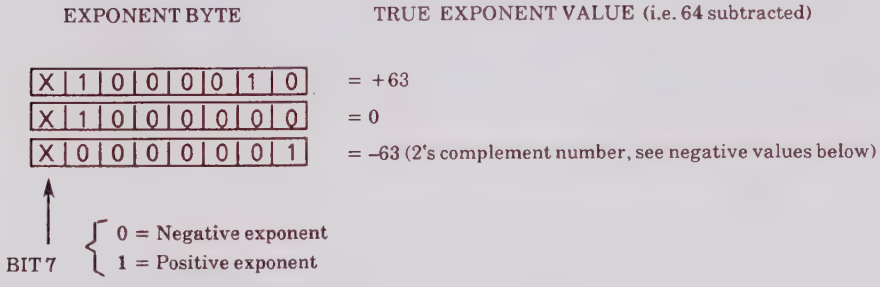
The exponent byte contains the sign bit of the result (the msb) and the exponent value expressed as an Excess 64 number. To obtain the true exponent value, subtract 64.

eg if the Exponent byte was



1. First take off the sign bit. The value then becomes: 1000010
2. Next, subtract 64 from the number; the value then becomes: 0000010
   Since $0000010_2 = 2$, the true value of the Exponent is 2. The exponent value (n) acts as a multiplier (2^n) on the mantissa.

The exponent value can be in the range +63, 0 and –63. The bit patterns in the Exponent byte for true values of +63, 0 and –63 are shown as follows :-


EXPONENT BYTE
TRUE EXPONENT VALUE (i.e. 64 subtracted)
X 1 0 0 0 0 1 0
= +63
X 1 0 0 0 0 0 0
= 0
X 0 0 0 0 0 0 1
= –63 (2's complement number, see negative values below)
BIT 7
0 = Negative exponent
1 = Positive exponent

When the entire Exponent byte is zero, ie 0 0 0 0 0 0 0 0 , the number (exponent + mantissa) is assumed to be zero.

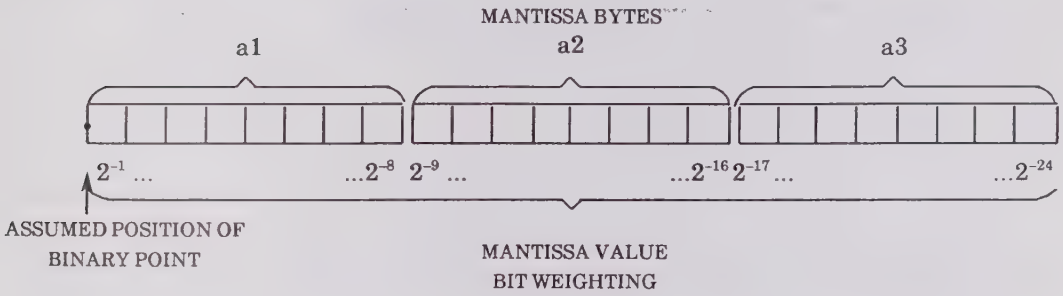
Negative values A negative exponent is expressed in 2's complement form. To convert the bit pattern of a negative exponent (ie bit 7 = 0); first subtract 64, then apply 2's complement. This gives the positive equivalent of the negative number.

eg An exponent of – 7 is given by the following bit pattern: X 0 1 1 1 0 0 1

Subtracting 64 from 0111001 gives 1111001. Applying 2's complement to 1111001 gives 0000111. This equals 7, indicating that the exponent value is –7.

Mantissa Byte

The mantissa is a floating point, non-normalised value with no hidden bits. The bytes a1, a2 and a3 should be considered as one long byte which contains the mantissa value. The bit weighting and assumed position of the decimal point is as follows:-



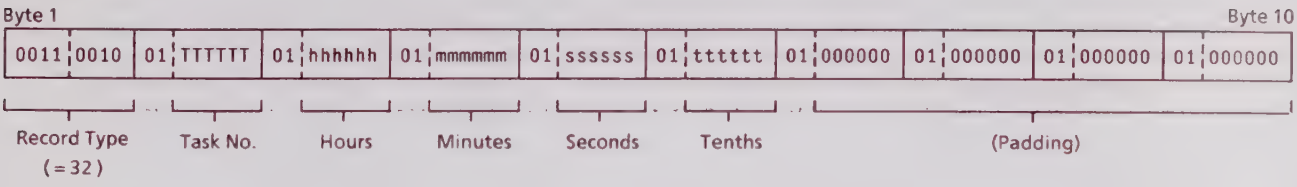
Mantissa polarity is given by the msb of the Exponent byte, 1 = -ve, 0 = + ve.

The effect of the exponent value (n) is to apply a multiplication factor of 2n to the mantissa value. (This has the same effect as moving the decimal point and all bit weightings; ie, a positive exponent n will move the decimal point n places to the right , a negative exponent n will move the decimal point n places to the left).

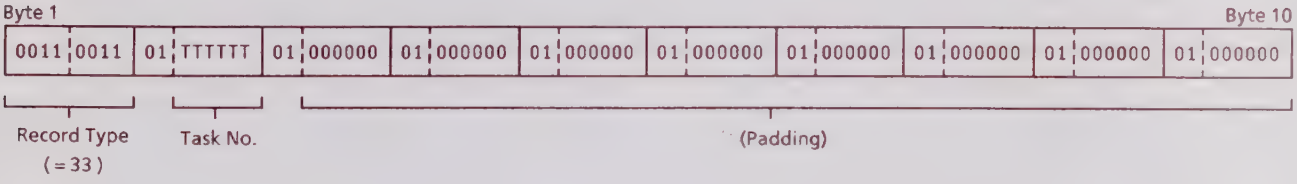
The exponent byte and the Mantissa bytes are encoded into byes 5 to 10 (inclusive) of the Numerical Result.

3.2 START AND END OF SCAN MARKERS

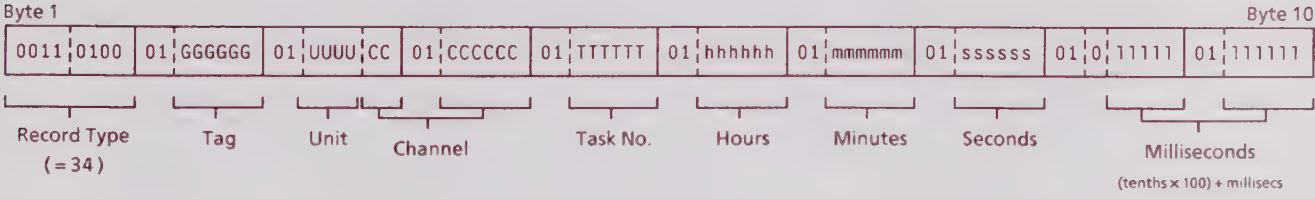
START MARKER



END MARKER

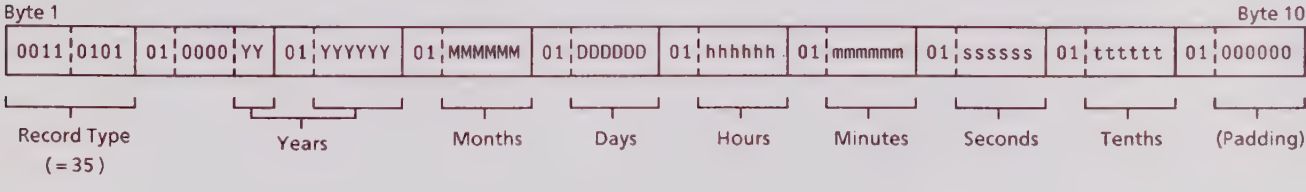


3.3 EVENT RESULT

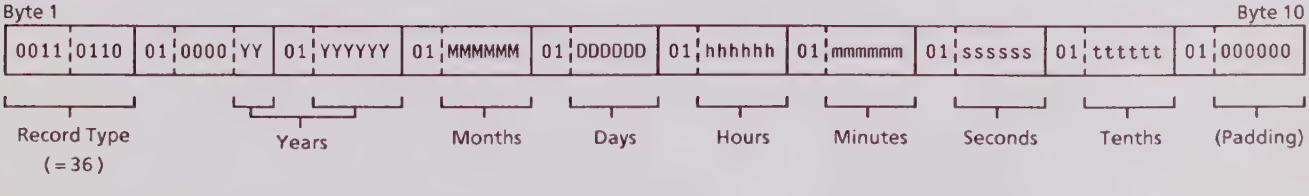


3.4 RUN AND HALT MARKERS

RUN MARKER

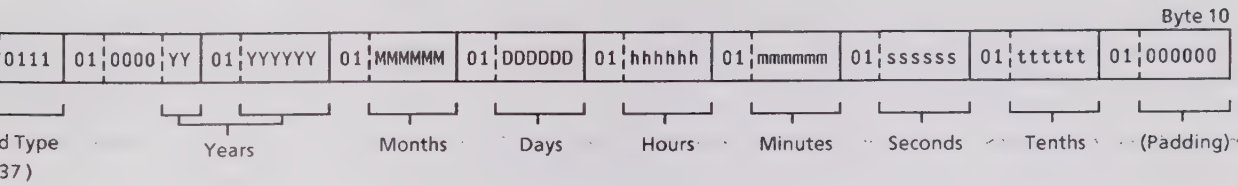


HALT MARKER

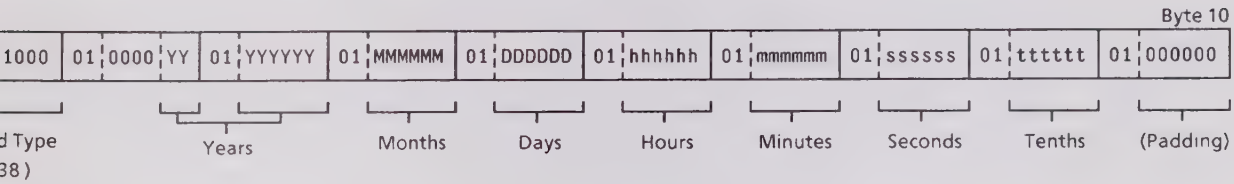


3.5 POWER FAIL AND POWER BACK MESSAGES

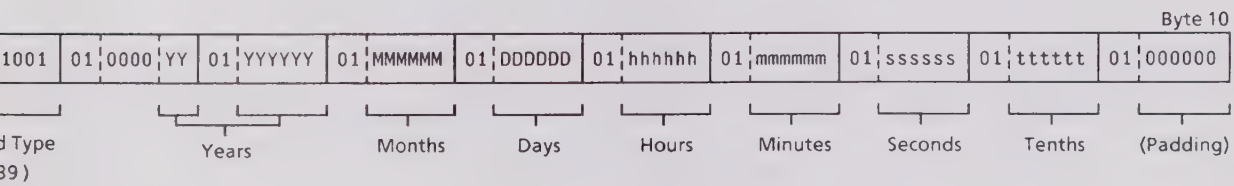
BACK



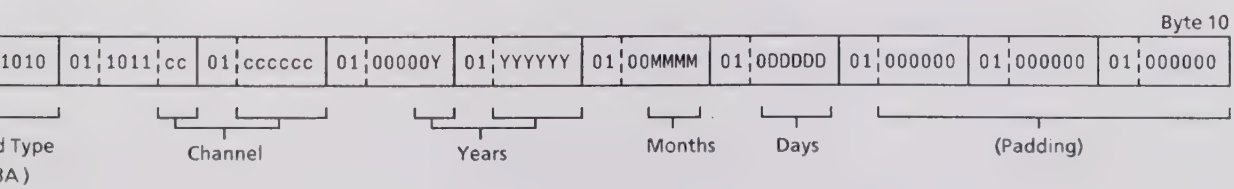
FAIL



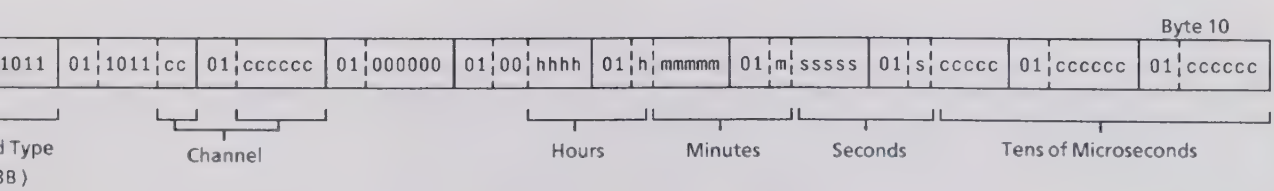
3.6 DATE



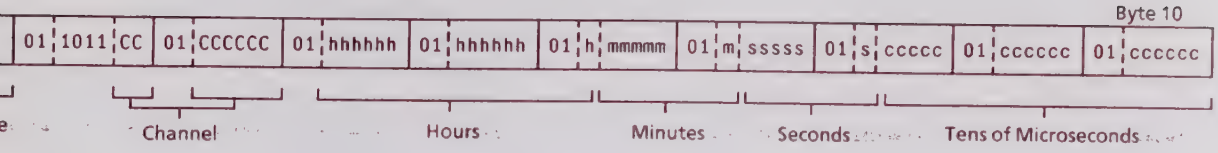
3.7 TRANSIENT START DATE MARKER


Byte 10
1010 | 01 1011 cc | 01 cccccc | 01 00000Y | 01 YYYYYY | 01 00MMMM | 01 0DDDDD | 01 000000 | 01 000000 | 01 000000
d Type
3A)
Channel
Years
Months
Days
(Padding)

3.8 TRANSIENT START TIME MARKER

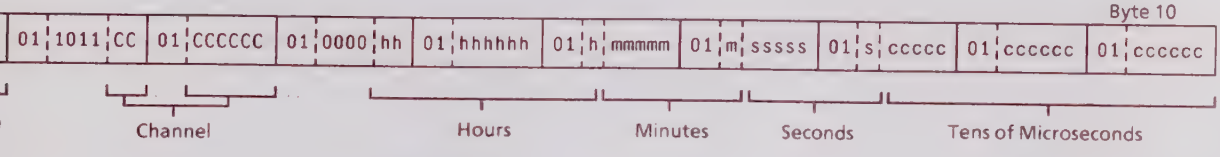

Byte 10
1011 | 01 1011 cc | 01 cccccc | 01 000000 | 01 00 hhhh | 01 h mmmmm | 01 m sssss | 01 s ccccc | 01 cccccc | 01 cccccc
d Type
3B)
Channel
Hours
Minutes
Seconds
Tens of Microseconds

3.9 TRANSIENT TRIGGER DELAY TIME MARKER

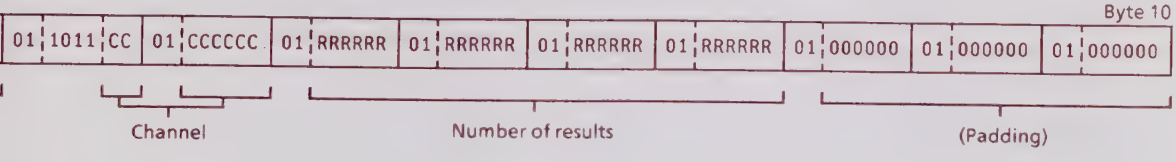


* Note that the hours are offset by 467 hours.

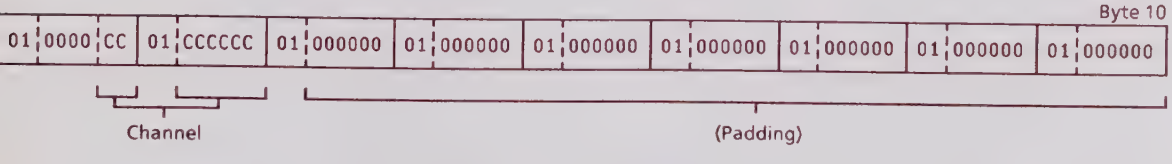
3.10 TRANSIENT INTERVAL TIME MARKER



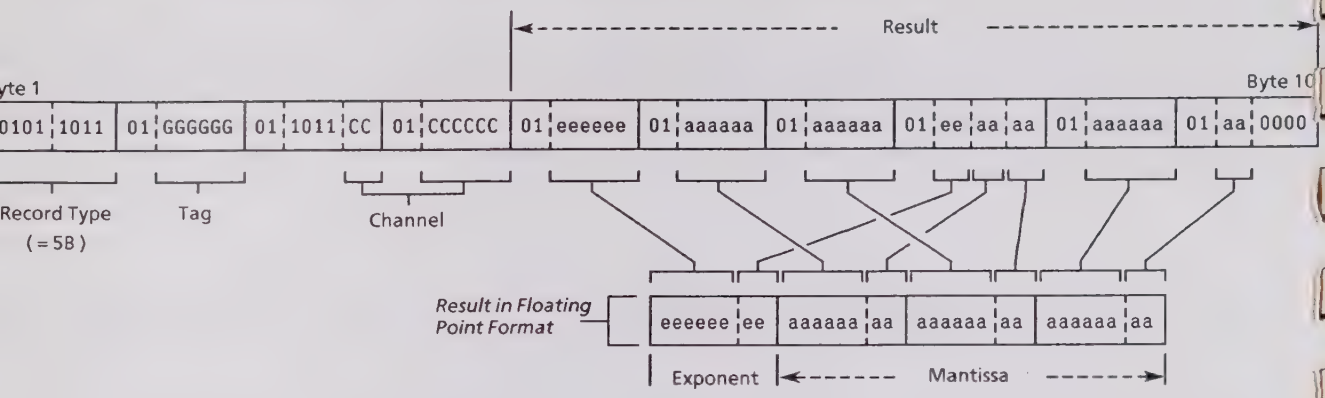
3.11 NUMBER OF RESULTS IN CAPTURE MARKER



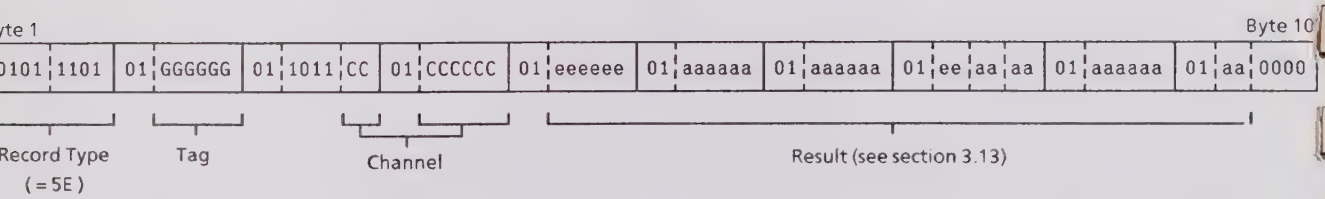
3.12 TRANSIENT END OF EXPORT MARKER



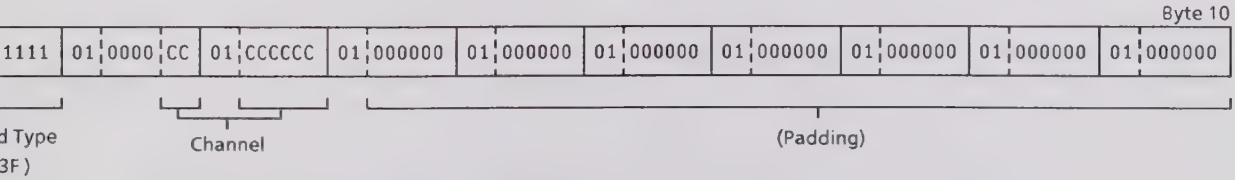
3.13 TRANSIENT CHANNEL MAXIMUM RESULT MARKER



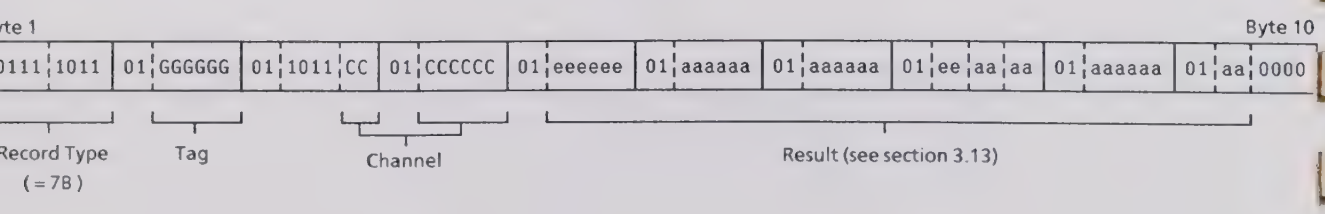
3.14 TRANSIENT CHANNEL MINIMUM RESULT MARKER



3.15 TRANSIENT EXPORT ABORT MARKER



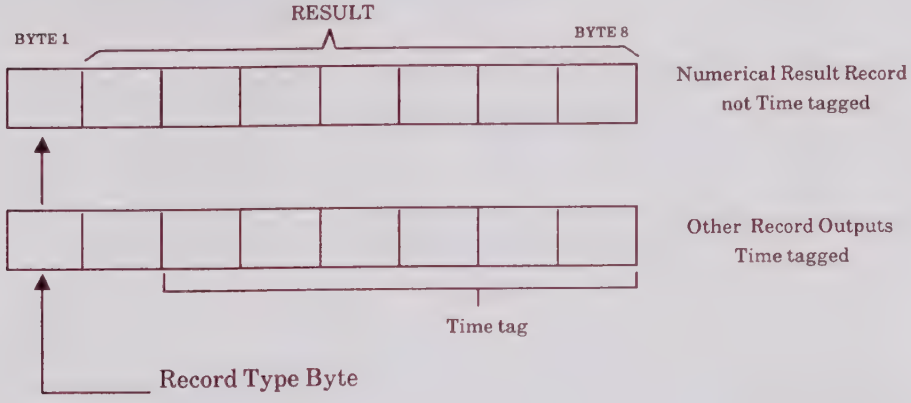
3.16 TRANSIENT CHANNEL RESULT



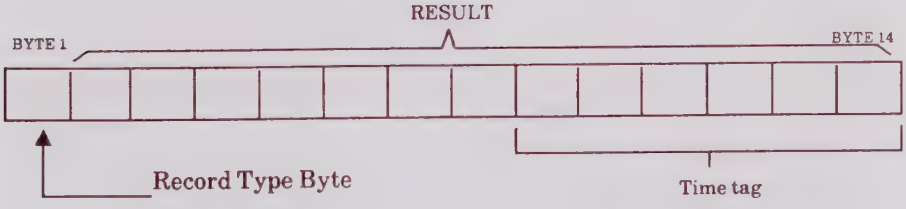
4 BINARY FORMAT

Record output in Binary format consists of 8 or 14 byte packets. Apart from the Numerical Result, all 8 byte packets are time tagged. The content of the bytes depend on the record-type and does not necessarily result in a printable character.

Binary Format, 8 Bytes



Binary Format, 14 Bytes



Record Type Byte :

The various types of Binary Format records are listed below.

[Ascii]	*[HEX]*	*Record Type*
'	60	Numerical Result not time tagged : normal logged result
a	61	Numerical Result not time tagged : monitored result
b	62	Start of Scan Marker
c	63	End of Scan Marker
d	64	Event Result
e	65	Run Marker
f	66	Halt Marker
g	67	Power Fail
h	68	Power Back
i	69	Date

[Ascii]	*[HEX] Record Type*		
j	6A	Transient Start Date	
k	6B	Transient Trigger Delay Time	
l	6C	Transient Interval Time	
m	6D	End of Export Record	
n	6E	Transient Channel Result	
p	70	Numerical Result (time tagged)	: normal logged result
q	71	" " "	: monitored result
t	74	Event Result (time tagged)	
y	79	Transient Start Time	
z	7A	Export Abort Record	
|	7C	Maximum Result in Capture	
}	7D	Number of Samples in Capture	
~	7E	Minimum Result in Capture	

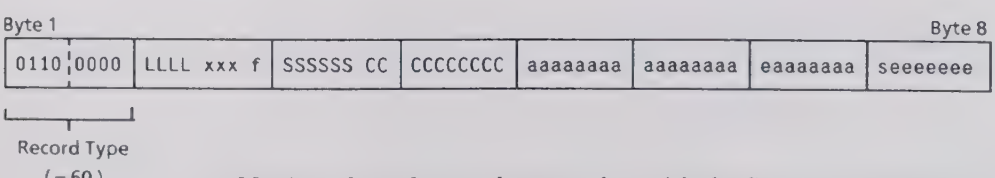
The binary data record contains all the numeric information that the equivalent full and compact format records contain. However, for certain record types, the binary record contains additional information not found in the full or compact record, ie:

a) The event record contains the Task number.
b) The Halt, Run, Power fail, Power Back and Date records have the year included in the date.

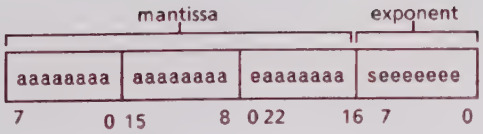
The following symbols are used in the format diagrams :

Symbol	Meaning	Range	Size (bits)
C	Channel number	1-600	10
T	Task number	1-8	4
a	IEEE mantissa		24
e	IEEE exponent		8
S	Status/error tag		6
s	Sign bit		1
L	Space for logger number to be inserted by external computer / controller	1-8	4
f	Digital result flag	0-1	1
x	Spare	-	-
n	Number of samples in export	-	-

4.1 UNTAGGED NUMERICAL RESULT



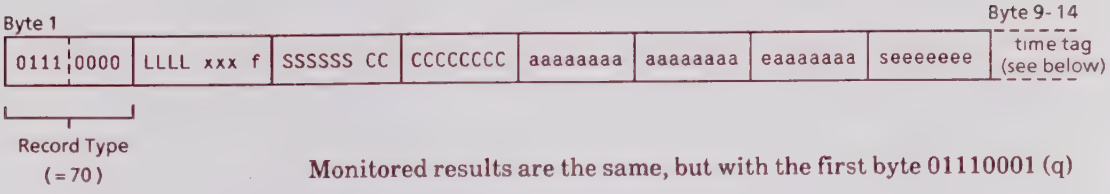
Monitored results are the same, but with the first byte 01100001 (a)



* Note : this applies to all similar formats that may follow :

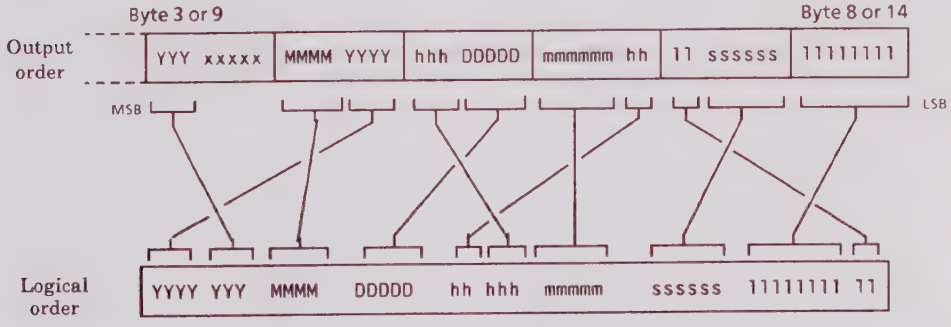
Number is :- $(-1)^s\ 2^{(\text{exponent}-127)} * (1 + \text{mantissa bits})$

4.2 TIME TAGGED NUMERICAL RESULT



Monitored results are the same, but with the first byte 01110001 (q)

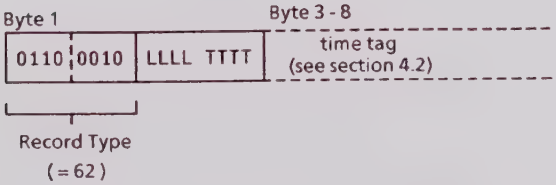
Time Tag



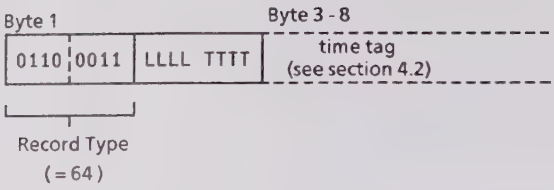
The Time Tag applies to all of the following format diagrams and the table below details the symbols used :

Symbol	Meaning	Range	Size (bits)
D	Day	0-30	5
M	Month	0-11	4
Y	Year	0-99	7
h	hours	0-23	5
m	minutes	0-59	6
s	seconds	0-59	6
l	milliseconds	0-999	10

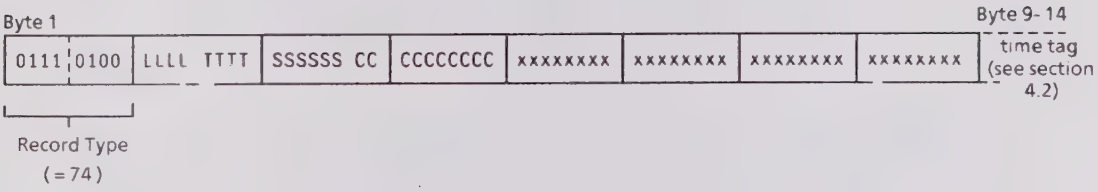
4.3 START OF SCAN MARKER



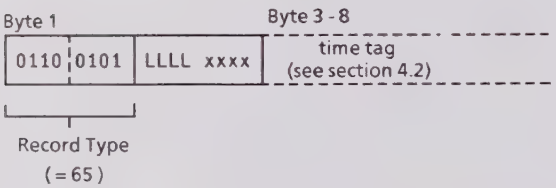
4.4 END OF SCAN MARKER



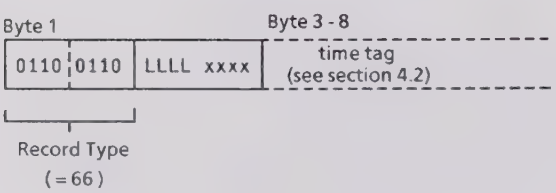
4.5 EVENT RESULT



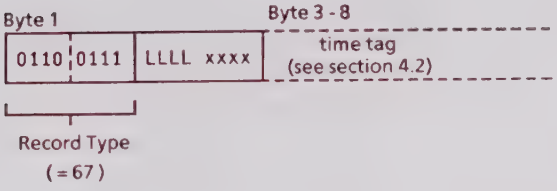
4.6 RUN MARKER



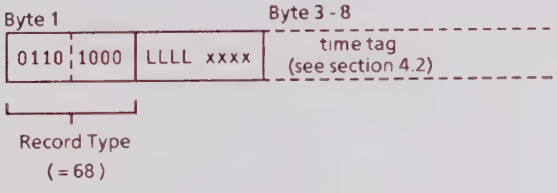
4.7 HALT MARKER



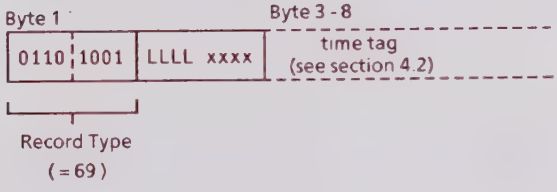
4.8 POWER FAIL MARKER



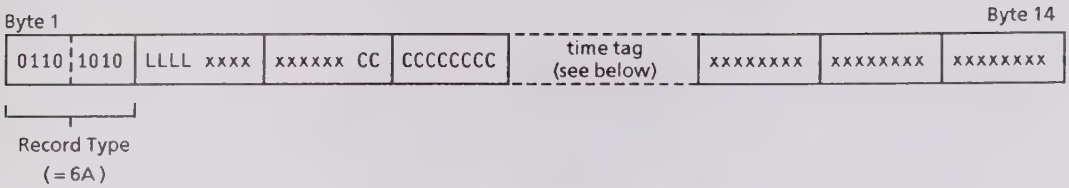
4.9 POWER BACK MARKER



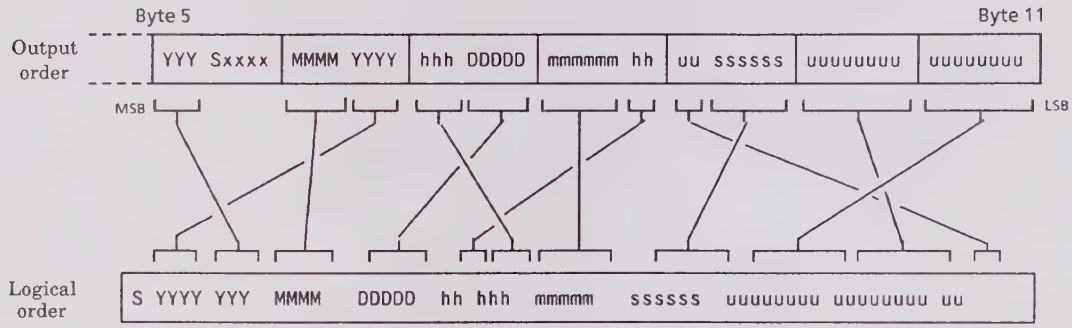
4.10 DATE



4.11 TRANSIENT START DATE



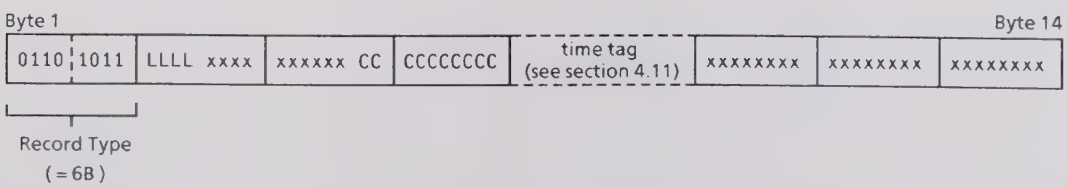
Time Tag



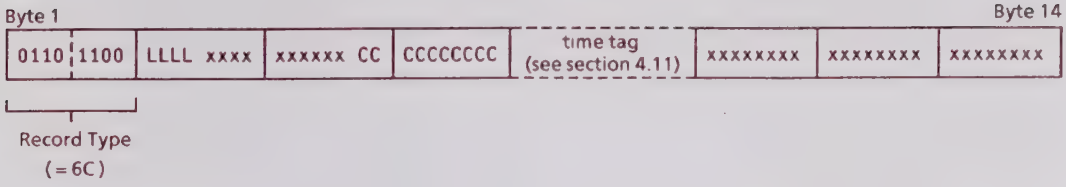
The Time Tag applies to all of the following format diagrams and the table below details the symbols used :

Symbol	Meaning	Range	Size (bits)
S	Sign	-	1
D	Day	0-30	5
M	Month	0-11	4
Y	Year	0-99	7
h	hours	0-23	5
m	minutes	0-59	6
s	seconds	0-59	6
u	tens of microseconds	0-99999	18

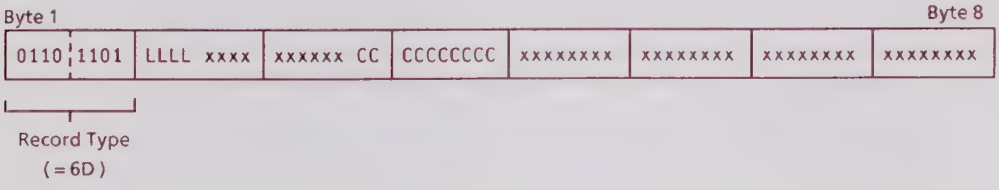
4.12 TRANSIENT TRIGGER DELAY TIME



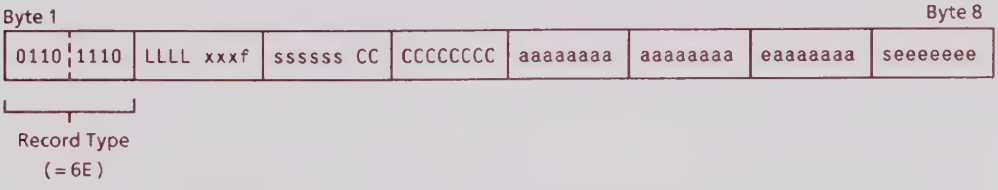
4.13 TRANSIENT INTERVAL TIME



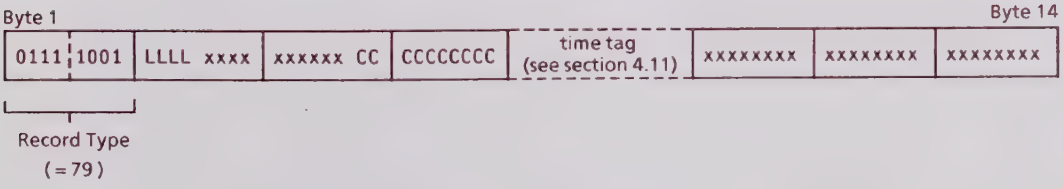
4.14 END OF EXPORT RECORD



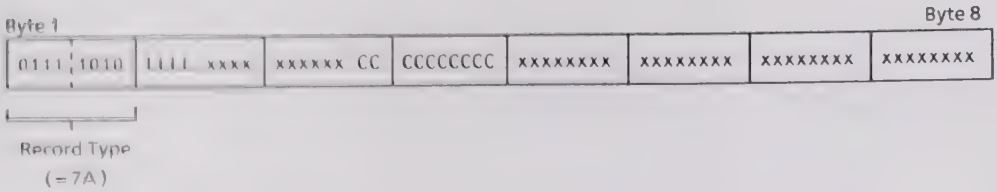
4.15 TRANSIENT RESULT



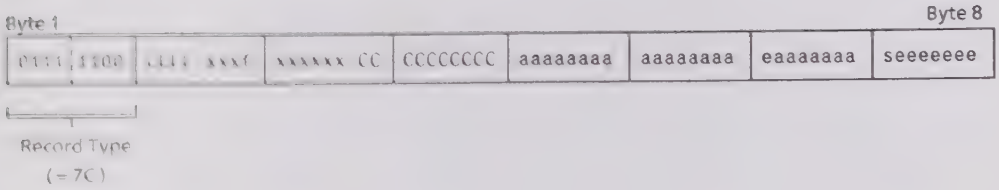
4.16 TRANSIENT CAPTURE START TIME



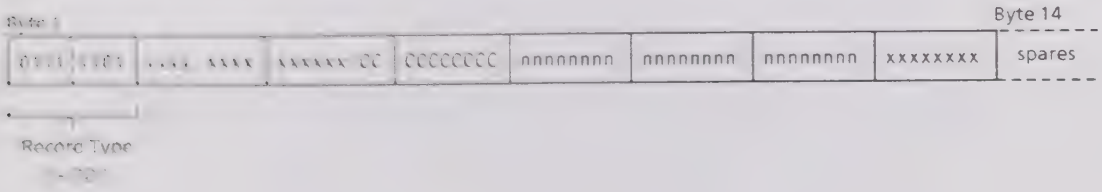
4.17 EXPORT ABORT RECORD



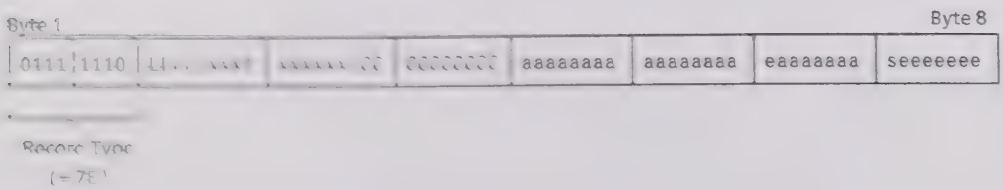
4.18 MAXIMUM RESULT IN CAPTURE



4.19 NUMBER OF SAMPLES IN CAPTURE



4.20 MINIMUM RESULT IN CAPTURE



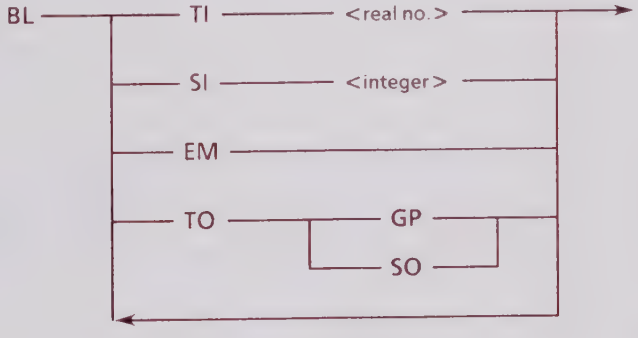
5 DATA BLOCKING

To improve the efficiency of communications with remote devices, SI3535 has the facility to block data. This means that data, normally sent as individual messages to a remote device, is retained in a data buffer (or block) within the instrument until the entire block is output.

An example of when it is preferable to output data in blocks is when the time required to gain access to a bus is large compared to the time required to transfer a single result. In this case it is more efficient to store several results and then request the bus once to transfer all the results in one block, rather than to request the bus repeatedly.

To select data blocking the SI3535 has a set of data blocking commands. When this command is given, data being output to either the Serial Output or the GPIB is stored in a data buffer until the conditions for outputting the buffered data are met. Output to any other device is unaffected by data blocking. Note that data blocking cannot be set up from the front panel menus of the SI3535.

The syntax of the data blocking commands is:



5.1 DATA BLOCKING COMMANDS

The function of each data blocking command is described below.

BLock TO GP
BLock TO SO
Data blocking can operate on only one output device at a time, and cannot work with the RS422 interfaces. Changing the output device causes the present data block to be output to the presently selected device immediately; the empty block then begins to fill with data for the new output device.

BLock TIme <Max. Time>
<Max. Time> is the maximum time that may elapse before an output must be given. It is expressed in seconds and tenths of a second, in the range 0 to 1987200 seconds (23 days). If 0 seconds is used, there is no maximum time limit. If the maximum time is exceeded, an output occurs when the next result is ready. This result is not output, but becomes the first result in the next block.

BLock Size <size>
<size> is the block size in bytes, from 1 to 8192. A block size of 1 turns data blocking off. If the block size is small, e.g. 1-7, and only one result can be stored in a block, then the results are output singly in block format. When a result arrives that would cause the block size to be exceeded then this result is excluded from the data block, which is now output. The excluded result becomes the first result in the next block. This ensures that the block size is never exceeded.

BLock EMpty
This commands the data block to output. It can be used when there is a likelihood that the data will not be output, or when you want to override the conditions controlling the output of data blocks.

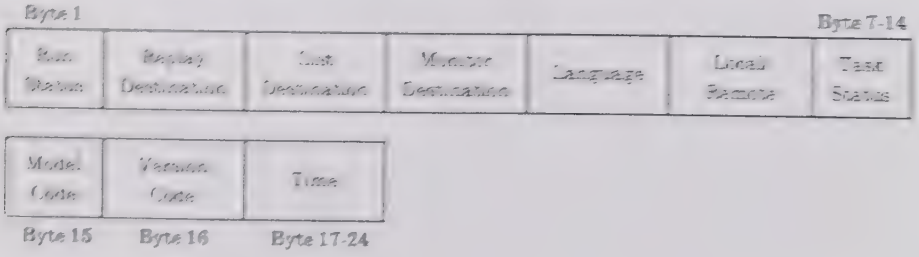
5.2 DATA BLOCKING OUTPUT CONDITIONS

There are a number of conditions which cause the data block buffer to output its contents to the output device.

1. The maximum block time has been exceeded and another result has occurred.
2. The block size would be exceeded by the latest result.
3. A *BLock EMpty* command has been received.
4. A *MOnitor OFf* command is received.
5. Data arrives for the output device from different sources (streams) e.g. from monitored data and task data. This could mean that output data blocks are quite short, because monitored data and task data will occur together. Other possible data streams are from disk, list and message data. Responses to commands will not cause the data block to be output.
6. When a Halt marker is detected. (This is to avoid data being lost if the Run command is never subsequently given.)
7. If the data blocking output device is changed whilst the logger is running.
8. If the previous output block was from the disk.

6 STATUS VARIABLES

If a query command (ST?) is sent to the logger, a single ASCII 24 byte status record is returned. This has the following format:



The individual bytes have the following ASCII codes :

Table 5.2 *Status Query Information*

Item	Values	Format	Size
Run Status	Run Halt Pause	R H P	1
Replay destination	Off GPIB Display Printer Serial 0,1,2 Disk Null	O G D P 0/1/2 K N	1
List destination	Off GPIB Display Printer Serial 0,1,2 Disk Null	O G D P 0/1/2 K N	1
Monitor destination	Off GPIB Display Printer Serial 0,1,2 Null	O G D P 0/1/2 N	1
Language	UK US French German	B A F G	1
Local/remote control	Local Remote	L R	1
Status of each task	Idle (0) or Triggered (1)	0/1 (×8)	8
Model code	D F G	D F G	1
Version	A (first launch)	A	1
Time	Time string	hh:mm:ss	8

7 STORING DATA ON DISK (SI3535D & SI3535G)

The number of SCORPIO measurements that can be written to disk will depend on the recording format of the measured data, Coded, Compact or Full, and the number of slots reserved for SCORPIO disk programs.

7.1 NORMAL/CYCLIC STORAGE MODES

The mode for recording measured data is selectable and can be 'Normal' or 'Cyclic'.

a) **Normal mode:** blocks of data are written to disk until SCORPIO detects that the disk is full. Data that is output to a full disk is stored temporarily in a 5000 measurement result buffer. When the disk drive is ready the stored results are output to it automatically.

b) **Cyclic mode:** data is written on to the disk until the disk is full, at which point SCORPIO will continue saving data by over writing previously saved data. This ensures that the disk always contains a record of the latest data.

   Although with transient it is not advisable to use cyclic mode. The reason for this is that when the disk is full and starts to overwrite previously saved data, the data header is lost thus losing the main information of the results recorded.

Note: Any program space that has been allocated on the disk will remain, and will not be overwritten.

7.2 DISK FORMATS

The measurement data will be stored on a disk which is formatted to one of two standards: HP-LIF or IBM MSDOS. The data will be written as 8 bit ASCII characters in one of three formats: Full, Compact or Coded (Binary format cannot be written to or replayed from the logger disk). The number of measurements saved in these formats differ between HP-LIF and MSDOS.

7.2.1 Compact and Full Format

Disks recorded in Compact or Full Format can be replayed in any suitable read station without the need for any decoding. For HP-LIF, one disk can accommodate a maximum of 28,625 measurements in Compact form or 14,995 measurements in Full format. The equivalent values for MSDOS are 33,175 measurements in Compact format or 17,375 measurements in Full form.

7.2.2 Coded Format

If SCORPIO processing has not been used, each 10 character ASCII string will contain numeric data in binary form, encoded into printable ASCII characters. Up to 52,480 results in HP-LIF and 60,820 results in MSDOS can be accommodated in this form.

However, if processing has been applied, under certain conditions, each data record will be followed by a second 10 character record that will contain pointers to the SCORPIO data areas and will be used by SCORPIO when reconstituting previously stored data. The amount of data that can be stored on disk may therefore be reduced by up to 50%.

Disks recorded in Coded format would normally be replayed in SCORPIO, but can be replayed in other read stations if suitable decoding is applied to the data. However, if the disk contains processed data the records containing pointer information should be ignored as they contain no meaningful information for the user. The pointer record may be identified by a channel number of zero.

Note: When using SCORPIO to decode a previously recorded disk containing processed data, the program under which the disk was recorded must be reloaded into SCORPIO. This program must be identical in every respect to the original program in order to obtain meaningful results.

7.3 DATA OUTPUT FORMAT

The data records are output to the disk in MSDOS or HP-LIF format. The data stored on the disk is stored in the file SIDASDAT.DAT when MSDOS format is used and in the file SIDASDAT in the case of HPLIF.

The data record type may be identified however, by the first character of each string. See section 2 for details.

In addition to measured data the disk may also contain a copy of the complete SCORPIO database.

7.4 SCORPIO DISK PROGRAM

An SCORPIO disk program consists of a complete copy of the SI3535 D & G database. It will contain all the channel set up information, task definitions, processing programs, etc., that were in the SCORPIO database when the save command was given, ie., the program may contain information for up to eight tasks and associated processing programs.

Space must be reserved on disk for each program to be saved, a disk can contain a maximum of 3/4 programs. The space reserved for each disk program will be 173kbytes and the programs will be stored in the file SIDASPRG.BIN for MSDOS and SIDASPRG for HP-LIF.

7.5 DISK COMPATIBILITY

Programs recorded on 3531D or G series data loggers can be converted for use on SI3535D,F or G data loggers.

This can be done using the "Orion Program Disk Translator" program running on an IBM PC. Contact your local Schlumberger dealer for details of this software, which is free of charge. The two application notes available are:

Orion-Scorpio Program Translator	Reference P/3535/01	Issue AA
Data to DIF data conversion	Reference P/3531/01	Issue A

7.6 HOW TO REPLAY THE DISK IN ANOTHER MACHINE

7.6.1 MSDOS Format

MSDOS disks can be replayed easily and without any conversion programs.
On a MSDOS disk each 'record' (with start task marker, line of results, halt marker, etc.) is delimited with CR/LF at the end of the line.

```
eg.     C 001 1.23456 Vdc  <CR><LF>
        HALT               <CR><LF>
```

Note: All records must be 10,20,30,40,50,60,70 or 80 characters long. (They are padded out with spaces if necessary.)

7.6.2 HPLIF Format

HPLIF disks can be replayed with the following HP BASIC program :

```
5   DIM Data$[80];
10  ASSIGN @File TO "SIDASDAT"
20  ENTER @File;Data$
30  PRINT Data$
40  ON END @File GOTO 60
50  GOTO 20
60  ASSIGN @File TO *
70  END
```

On HPLIF disks each reord has two bytes at the start describing its length.

```
eg.     <00><28> C001   1.234567 Vdc   C002   2.345678 Vdc
        <00><06> HALT
        <FF><FF>
```

Note: The above lengths are in hexidecimal and the value FFFF indicates the end of the file.

Chapter 6
Serial Interface

Section		*Page*
1	**General Information**	6.3
2	**Standard RS232 Interface**	6.4
2.1	Computer Operation	6.4
2.1.1	Result Transfer Mode	6.4
2.1.2	Control Mode	6.4
2.1.3	Relinquishing Control	6.7
2.1.4	Communication Protocol	6.8
2.2	VDU Printing Terminal Operation	6.8
2.2.1	Result Transfer Mode	6.8
2.2.2	Control Mode	6.8
2.3	Output Only Protocol	6.12
2.4	Connection to RS232 Port(s)	6.12
3	**Fast Serial Interface**	6.16
3.1	Output Porting	6.16
3.2	Communication Protocols	6.16
3.2.1	BS4505 Protocol	6.16
3.3	Connection Details	6.19
3.3.1	Cable for RS422 Operation	6.19
4	**Front Panel Emulation**	6.24

Tables

6.1	RS232 Command and Response Summary	6.11
6.2	RS232 Serial Interface Pin Details	6.13
6.3a	Controller Front Panel Switsh Settings - SW2	6.13
6.3b	Mode Selection - SW2	6.14
6.3c	Language Selection - SW2	6.14
6.4a	Controller Transmit/Receive Speed Selection	6.15
6.4b	Controller Data Format Selection	6.15
6.5	Serial Data Header Characters	6.17
6.6	Serial Interface PL1/PL2 Pin Details	6.21
6.7	Switch Settings for Serial Interface Module	6.23

Illustrations

6.1	General Peripheral Interconnections	6.3
6.2	Main Control Loop	6.9
6.3	Keyboard Interrupt Routines	6.9
6.4	SI3535 Interrupt Service Routine	6.10
6.5	Location of PL1 and switches on the Controller module	6.13
6.6	Location of SW1 on the controller module	6.14
6.7	SI3535 BS4505 - Transmit Protocol	6.19
6.8	SI3535 BS4505 - Receive Protocol	6.20
6.9	Location of PL1 and PL2 on the Serial Interface module	6.22
6.10	Location of SW1-4 on the Serial Interface module	6.22
6.11	Front Panel Emulator Interconnection	6.24

1 GENERAL INFORMATION

This chapter provides details of the communication protocols and connection details for the serial data interfaces available with the SI3535. The serial interfaces are implemented as follows:

a) Standard RS232 interface on the Controller Module which is used for remote control of the SI3535 by a computer or terminal. Connection of output only recording devices is also possible.

b) Serial Interface Module (35304B) with 2 output ports, which can be set independently for either RS422 or RS232 operation.

c) A RS232 connector is available on the front panel of SI3535 F and G for Front Panel Emulation using VT220 or compatible.

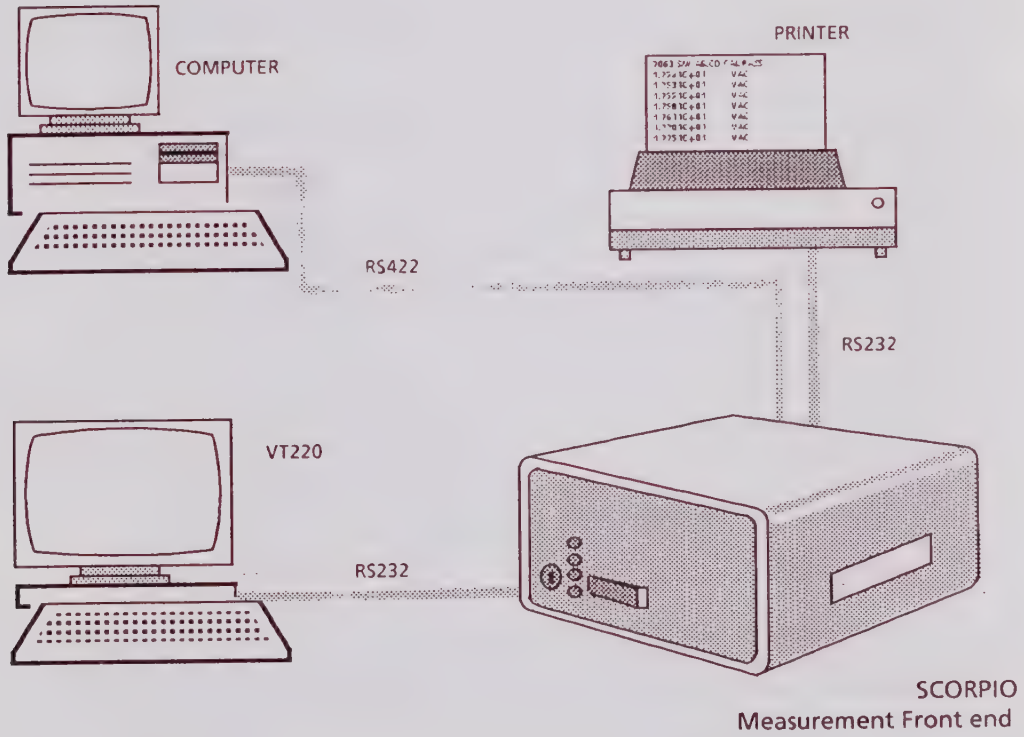


Fig 6.1 *General peripheral interconnections*

2 STANDARD RS232 INTERFACE

2.1 COMPUTER OPERATION

When using the standard RS232 interface with a computer, the following modes of operation are available:

a) Transfer of results from the SI3535 to the computer.

b) Control of the SI3535 by the computer for set-up, run/halt or to request specific information from the SI3535.

The two modes are mutually exclusive, such that if the computer is controlling the SI3535, results cannot be transferred.

2.1.1 Results Transfer Mode

Whilst the SI3535 is running tasks, the results for each channel are input to a buffer memory. If the RS232 interface is designated as the output port, the SI3535 can provide transfer of data, as soon as the first result is in the buffer

Results transfer is initiated by the SI3535 sending O to the computer, which must respond with ?, and the result is output. Successive interchanges of O and ? enable further results to be transferred to the computer.

Since results can be output to the computer at any time, the computer must include at least a single-character buffer which is periodically checked to see if the SI3535 has sent O. Checking of the computer buffer must occur frequently enough, to ensure that the SI3535 buffer does not fill up and prevent continuation of the current task.

In order to receive results, it is not necessary to set-up or control the SI3535 from the computer (this can be performed at the front panel for SI3535D versions, or from a local terminal for SI3535F and G). The computer effectively acts as an output peripheral device, receiving results by means of the O and ? routine If the computer has been used for control, it must release the communications link, allowing transfer of results to occur; this is achieved by sending the # character.

2.1.2 Control Mode

Under computer control, it is possible to set-up and modify the SI3535 database. The computer can also command the SI3535 to monitor channels or tasks, or perform an immediate channel measurement. To exercise control, the computer occupies the communications link, by sending character ! to the SI3535. The link is then no longer available for the transfer of results.

Note: The request (!) to establish the communications link will not be successful, if there is an outstanding prompt for data available (ie. If the SI3535 has sent O, data ready, for which it has not received a request for transfer, ? it will ignore the request to establish the link (!) until the O has been acknowledged, and the output read).

In reply to a computer request to establish the link, the SI3535 responds with one of the following status prompts, defining the level of control that the computer has over the SI3535:

N The SI3535 does not recognize the computer as a remote device for control purposes. The computer must identify itself by sending the **%1C/R** command in order to gain full control.

L The SI3535 recognizes the computer as a potential controller, but is currently in **LOCAL** control. The computer must send the **%4** or **%6** commands to gain full control.

C The computer has full control of the SI3535.

P The SI3535 recognizes the computer as a potential controller, but currently the computer cannot attain full control because a higher priority device is already in control.

Note: The standard RS232 interface has the lowest priority of the SI3535 remote interfaces.

2.1.2.1 Acquiring Control of the SI3535

Initially, with the computer connected and switched-on, the SI3535 will normally be in LOCAL mode. On establishing the link, the SI3535 responds with the N status prompt; to gain full control, the computer sends command %1CR, identifying itself to the SI3535 as a potential controller. The expected response would be either L or C prompts. However, if a higher priority device has already taken control, the prompt is P. In response to the L or C prompts, the computer can send either of the following commands to gain full control:

%4C/R Requests full remote control of the SI3535 if the response was L. The SI3535 normally responds with C.

%6C/R Puts the SI3535 into local lockout, and requests full control (if not already available); the expected response from the SI3535 is C. The lockout condition prevents access to the SI3535 from either the SI3535D front panel, or the SI3535G from a local terminal.

NOTE: When the communications link has been established, the computer has limited control over the SI3535. However full control is only established when the SI3535 replies with the C prompt.

2.1.2.2 Limited Control

When the computer has occupied the communication link by sending ! it has a limited control that may be exercised whichever response (**N,L,C** or **P**) is received from the SI3535. In limited control, the following can be actioned:

a) **Measure**: Demand a single immediate measurement from any one channel.

b) **Query**: Obtain data from a specified part of existing channel or task databases.

For both Measure and Query, the SI3535 checks the validity of the command and sends E or R, which are defined as follows:

E Command in error, the SI3535 is ready to send error information.

R Command has been actioned and the SI3535 is ready to send data.

On receiving E or R, the computer sends ? and the SI3535 outputs the data as a string of up to 80 characters terminated by C/R. When the data is an error code it will be of the form ExxxC/R, where xxx identifies the type of error. The SI3535 will not resend the N,L,C or P prompt if an error (E) or response (R) string was transmitted. Commands may be sent to the SI3535 as soon as the pending output has been read.

2.1.2.3 Full Control

When the response from the SI3535 is C, the computer has full control In addition to Measure and Query (available with limited control), the computer can also

a) Program: Set up and modify channel and task databases, run, halt, etc.

b) Monitor: Request monitor of specific channels or tasks.

c) List: The SI3535 sends the computer the information held in specified channel and task databases. To use this the computer would have to output the list to an associated VDU or printer, or store the list for subsequent output.

d) Read Disk: For SI3535D and SI3535G only. Programs and/or data which are stored on the SI3535 disk, can be read to the computer.

For Monitor, List and Read disk, data is available only after the computer has sent a # and is then transferred using the O and ? routine.

2.1.2.4 Communicating with the SI3535

When the link has been established and the computer has received the SI3535 status prompt, it should reply by sending either:

a) 'command string' C/R to communicate with the SI3535

b) or, if the link between the SI3535 and the computer is to be terminated, the character #

The type of command that can be sent to the SI3535 depends on whether the computer has full or only limited control.

When the reply is a 'command string', the SI3535 will check its validity before responding with the prompt:

E,R,N,C,P or L defined as follows:

E Command input in error, the SI3535 is ready to output error information.

R Command accepted, the SI3535 is ready to output data in response to the command.

N,C,L or P Command accepted, the SI3535 is ready to accept the next command.

If the E or R prompt was received, the computer must respond with a question mark:

?

The SI3535 sends up to 80 characters of response data followed by a carriage return (C/R). If the response was an error message it will be in the form:

E nnnC/R

nnn is the numerical error code.

Following any response/error data, the SI3535 is ready to receive further command data or a hash (#) to terminate control mode input.

2.1.3 Relinquishing Control

The computer can relinquish control of the SI3535 in one of three ways as follows:

a) # By sending # the computer releases the communications link and allows the SI3535 to provide (O) results of measurement or other information that the computer may have requested. The # does not cancel the control status already set up by the computer; this status is regained if ! is sent.

b) %5C/R This tells the SI3535 to release local-lockout and assume local status. For SI3535D the front panel has control; with SI3535F or G the local terminal is able to gain control. The computer retains the RS232 link and may regain full control by sending %4 or %6. Results cannot be output unless # is sent.

c) %3C/R The computer relinquishes the status of controller, but if the SI3535 is in local lockout this command does not remove it. The status reverts to N allowing another remote device to take control, and any intervention at the front panel or local terminal is precluded. Results cannot be output unless # is sent, since the computer still occupies the RS232 link. Full control may be regained by sending %1, followed by %4 or %6 as required.

2.1.4 Communications Protocol

The flow diagrams in Figures 6.2, 6.3 and 6.4 give examples of the communications protocol required to communicate with the SI3535 when using an interrupt driven computer, such as an IBM PC.

The main control loop in which the program resides while waiting to send a command to, or receive data from, the SI3535 is shown in Figure 6.2. Two keyboard interrupt routines, one that sends a character to the SI3535 in order to gain control of the communications link and a second that sends a character to the SI3535 to terminate the link. Figure 6.3 shows the SI3535 interrupt service routine that checks the validity of the prompt character received from the SI3535 and takes the appropriate action.

For computers, which do not support interrupts, it will be necessary to monitor the input buffer and keyboard buffer for input. Monitoring should take place at frequent intervals so that prompts are not missed. The interval between receiving the last character of the SI3535 data, and the next prompt being set, is dependent on the amount of processing being carried out by the SI3535, and transmission baud rate selected.

Note: The prompt could be sent immediately following the stop bit of the final data character; however in practice there is normally a short delay.

2.2 VDU/PRINTING TERMINAL OPERATION

As for the operation of the SI3535 with a computer (Section 2.1), both the results transfer and control modes are possible. The protocol used for a VDU or printer with keyboard is similar to that for the computer.

2.2.1 Results Transfer Mode

Data is output to the terminal immediately it becomes available. However, data cannot be displayed or printed while the terminal is occupying the communications link. Any data sent while the communications link is established will be stored in the buffer. To receive data the link should be terminated by typing a # character at the terminal.

2.2.2 Control Mode

As detailed in Section 2.1 the communications link is established by typing an ! at the terminal and the prompts returned by the SI3535 are N,L,C or P. The response input at the terminal to gain full control of the SI3535 is %1, %4 or %6.

2.2.2.1 Limited Control

Measure and Query command may be used under limited control except that response data is output to the terminal immediately. The E and R and ? response and commands are not required under this protocol.

2.2.2.2 Full Control

All commands available under limited control may be used. In addition Program, List,Monitor and Disk commands may be used.

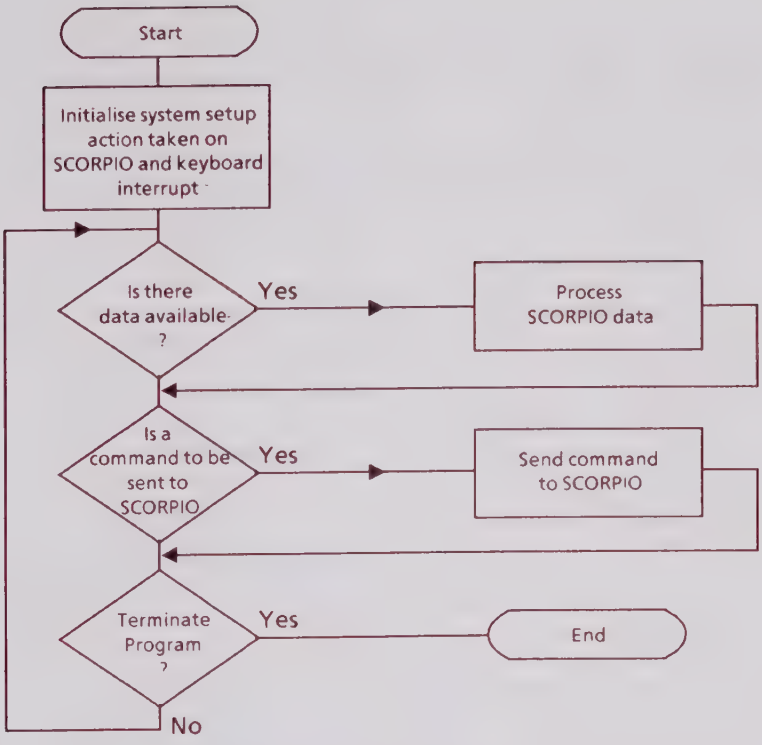
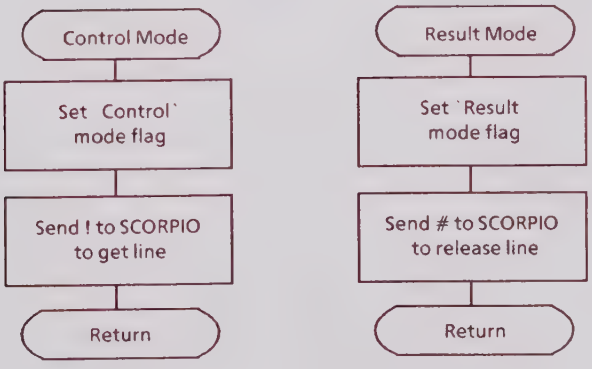


Fig 6.2 *Main Control Loop*



(a) *Request Control* (b) *Release Control*

Fig 6.3 *Keyboard Interrupt routines*

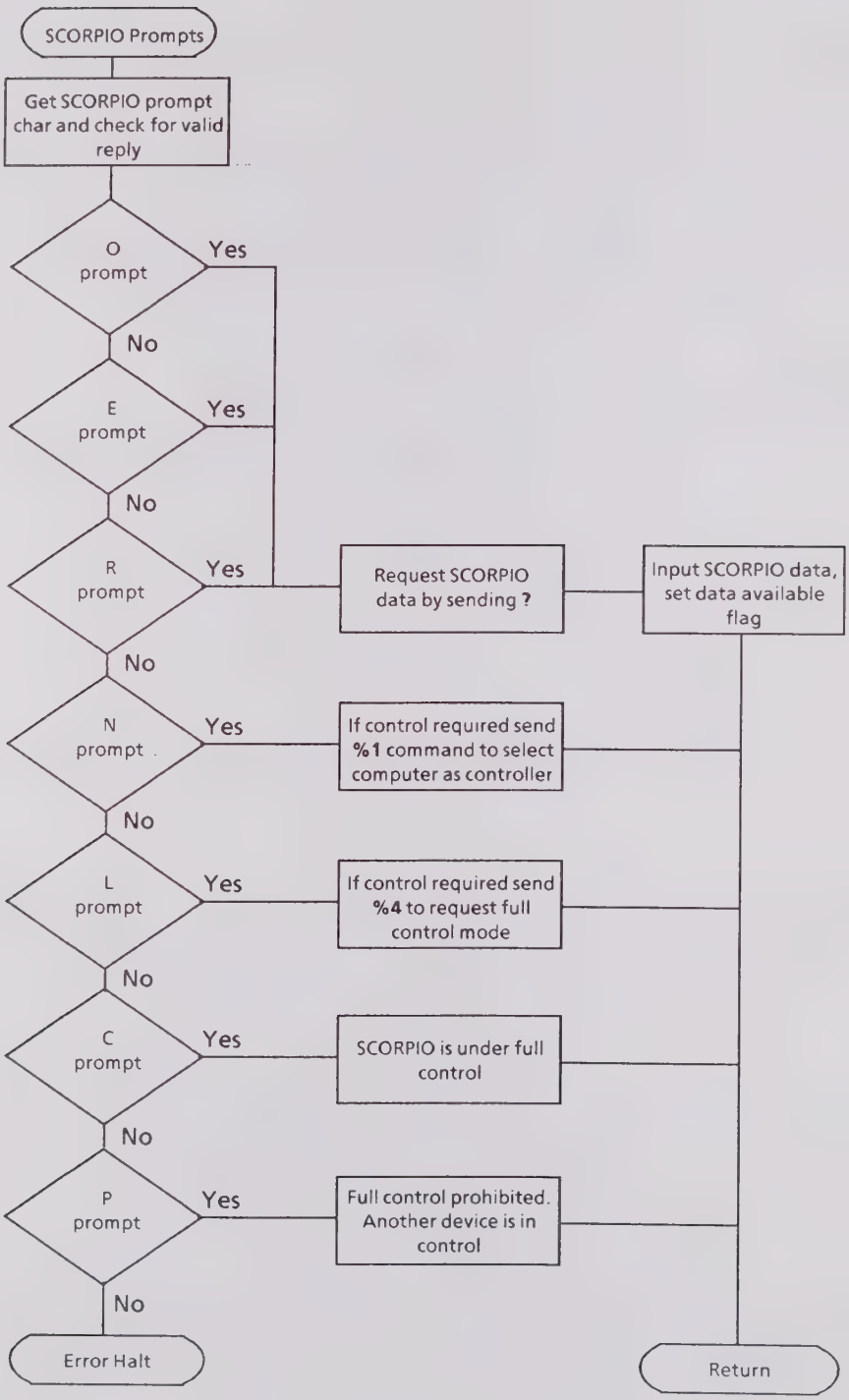


Fig 6.4 *SCORPIO Interrupt Service Routine*

2.2.3 Relinquishing Control

The communications link is terminated by typing a # character at the terminal. The %3 and %5 commands are used to relinquish remote status without terminating the control link.

Note: Typing # at any point in a command line will result in the link being terminated immediately and the command line being ignored.

Table 6.1 *RS232 Command and Response Summary*

Computer Input		SI3535 Response	
Code	Meaning	Code	Meaning
!	Attention. Computer occupies the communication link	N L C P	You (computer) have not identified yourself as a controller. I (SI3535) am in local, but you are recognized as remote controller. You have full control. High Priority device has control
%1C/R	If response was N. Computer identifies itself as a potential controller.	C L E	Computer has full control. The SI3535 is in local, but computer recognized as remote controller Error E113 the command has been input out of sequence.
%4C/R	If response was L, request full control.	C E	Computer has full control. Error E113 the command has been input out of sequence.
%6C/R	Assert local lock-out and go to local.	C E	Computer has full control. Error E113 the command has been input out of sequence.
%5C/R	Release local lock-out and go to local.	L E	The SI3535 in local. Error E113 the command has been input out of sequence.
%3C/R	I relinquish controller status, but not local lock-out	C E	Computer has full control. Error E113 the command has been input out of sequence.
Command String C/R	Send a command or series of commands to the SI3535.	C P E	Commands accepted. A higher priority device has taken control. Error E113 the command has been input out of sequence.

Table 6.1 *RS232 Command and Response Summary [continued]*

Computer Input		SI3535 Response	
Code	Meaning	Code	Meaning
Measure Command C/R	Request an immediate measurement.	R E	Result ready for output. Error in command input.
Command ? C/R	Query command	R E	Query data ready. Error in command input.
?	Request error number, immediate measurement data or query data	"Data" C/R	Error number in form *nnn* or upto 80 characters of query or measurement data
#	Release the link to permit transfer of results.		
RESULT TRANSFER MODE			
			Data ready to send
?	Request transfer of data	Data C/R	Up to 80 characters of result data. Followed by the O prompt when ready to send further data.

2.3 OUTPUT ONLY PROTOCOL

The Controller standard RS232 (PL1) port can be used to connect the SI3535 to external recording devices such as printers, cassette recorders or external disk drives. The SI3535 sends data to the output port as it becomes available. If the device is not ready to receive data, the data is lost (Table 6.2).

2.4 CONNECTION TO RS232 PORT(S)

For the connection of a terminal device to the standard RS232 connector on the SI3535 Controller, the correct baud rate and protocol must be selected. Details are given in the following tables .

The following components are located on the rear panel of the Controller Module (Figure 6.5):

PL1: Used for test purposes, and for connecting external RS232C-compatible devices.

The RS232C standard defines a single-ended bi-polar voltage circuit, capable of data rates up to 20 kB (the Controller can handle data rates up to 19.2 kB), using a maximum conductor length of 50 feet. This single-ended circuit uses a single conductor to carry each signal, with the voltage referenced to a common return conductor.

The Controller implements a restricted subset of RS232C (Table 6.3a).

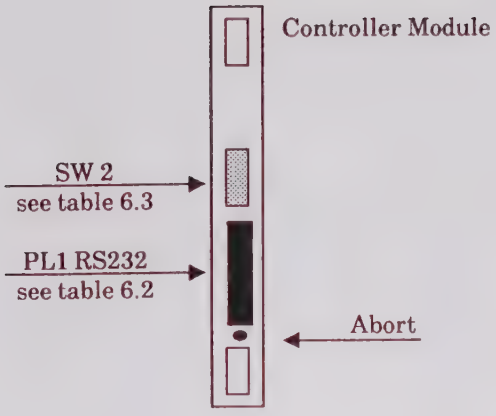


Fig 6.5 *Location of PL1 and switches on the Controller Module*

Table 6.2 *RS232 Serial Interface Pin Connections*

Pin Number	Function
2	TX Data
3	RX Data
4	RTS (Request to Send)
5	CTS (Clear to Send)
6	DSR (Data Set Ready)
7	Signal Ground
8	DCD (Data Carrier Detect)
20	DTR (Data Terminal Ready)

Table 6.3a *Controller Front Panel Switch Settings - SW2*

Switch Number	Function	Switch Position	
		OFF	ON
1,2	Mode select	See Tables 6.3b , 6.3c	
3,4	Language select		
5	Test	Normal	Test mode
6	Boot	-	Autoboot
7	Modem	No modem	Modem connected
8	Battery backup	No memory backup	Clock & memory backed up

Table 6.3b *Mode Selection - Switch 2*

Switch Settings		Mode	Remote Echo
Switch 1	Switch 2		
ON	ON	Computer protocol. CR termination.	N
OFF	ON	Terminal termination (CR,LF). No protocol.	Y
ON	OFF	Output only. CR termination.	N
OFF	OFF	Printing terminal. CR,LF + 4 nulls termination.	N

CR = carriage return; LF = line feed.

Table 6.3c *Language Selection - Switch 2*

Switch Settings		Mode
Switch 3	Switch 4	
ON	ON	British English
OFF	ON	American English
ON	OFF	French
OFF	OFF	German

Controller Module

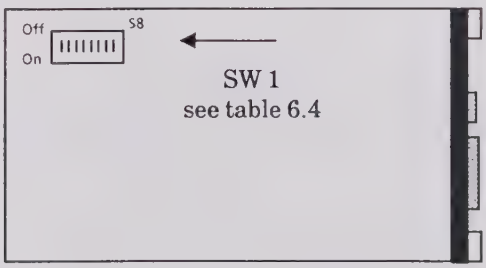


Fig 6.6 *Location of SW1 on the Controller Module*

Table 6.4a *Controller Transmit/Receive Speed Selection (SW1 BA1-BA3)*

Switch Settings			Transmit/Receive Speed (Bits/second)
BA3	BA2	BA1	
OFF	OFF	OFF	75
OFF	OFF	ON	300
OFF	ON	OFF	1200
OFF	ON	ON	9600
ON	OFF	OFF	19200
ON	OFF	ON	2400
ON	ON	OFF	4800
ON	ON	ON	19200

Table 6.4b *Controller Data Format Selection (SW1 S4-S8)*

Switch	Function Selected	
	Switch ON	Switch OFF
S4	Parity on	Parity off
S5	Parity even	Parity odd
S6	7 data bits	8 data bits
S7	XON/XOFF enable	CTS/RTS enable
S8	Reset on break	No reset on break

Note: Switch 8- "Reset on break" function, allows remote restart of the logger. It is intended for use with Modems where it is impractical to switch the machine off and on to restart the instrument.

It should, therefore, be set to OFF unless specifically required.

Care should be exercised if the switch is set to ON. This is because certain devices (particularly printers) issue a <BREAK> when turned off, which will cause the instrument to reset.

3 FAST SERIAL INTERFACE

3.1 OUTPUT PORTS

The 35304B Serial Interface Module has two output ports which can be set indepentently for either RS422 or RS232 operation. BS4505 has been fully implemented for both interface standards on Port 2; Port 1 is output only. In addition limited RS232 protocol is available to control external recording devices such as a printer or tape punch.

If Port 2 is used to control the SI3535, a request by this port for Remote status will take precedence over the standard RS232 port on the Controller module. Both Port 1 and 2 can be used for output only devices. A tracker unit or printer could be used on either port.

3.2 COMMUNICATION PROTOCOLS

The communications protocol defined by British Standard 4505 is primarily designed for use with equipment capable of being driven at high baud rates and/or over long distances via a RS422 balanced voltage connection (eg. a computer). However, the system can be used to RS232 operation by changing the connecting cables and selecting the appropriate data rate and operating mode. Note that when running BS4505 binary data cannot be output to RS232

The BS4505 protocol involves communication between two devices. The SI3535 is hardware configured so that it is always the lower priority device. Therefore if a request for control is made by the other (eg. a computer), that device will always gain control of the line.

3.2.1 BS4505 Protocol

For a device to transmit data, an enquiry (ENQ) is sent to the receiving device to establish control of the line. A positive acknowledgement (ACK) is then required by the receiving device to allow transmission. Once a device is in control it can send a data block with a format defined below:

SOH	Start of header marker.
HEADER	Characters indicating data type.
STX	Start of data marker.
DATA	Data block - max of 80 characters for the SI3535 protocol, excluding terminating characters (CR or CR/LF).
ETB	End of data block marker - when another block of data of the same type is to follow.
ETX	End of text marker - when this is the last block of data of this type and the line is to return to the neutral state.
BCC	Block check character.

Note: All characters are asynchronous, 7 data bits + 1 parity bit, even parity.

The receiving device checks the block of data for validity and if valid will acknowledge its receipt with an **ACK**, allowing a further block to be transmitted. If the received data is corrupt a negative acknowledgement, **NAK**, is returned to the transmitting device and the last block sent is re-transmitted.

Table 6.5 *Serial Data Header Characters*

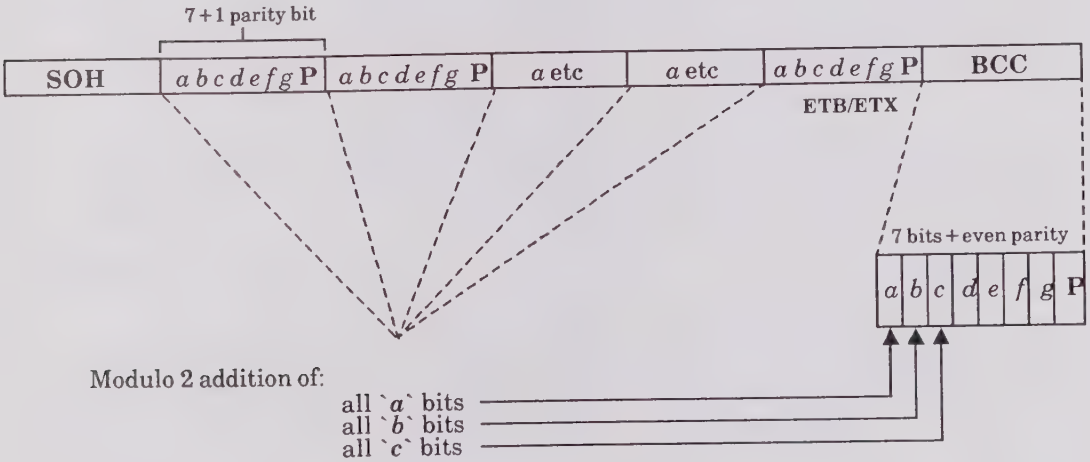
1st Byte (ASCII)	2nd Byte (ASCII)	Description of Data
0		Monitor data stream
	0	Monitor tak result
	1	Monitor channel result
	2	Monitor start marker
	3	Monitor end marker
	4	Monitor event
1		Logged data stream
	0	Ordinary logged result
	2	Start Scan marker
	3	End of Scan marker
	4	Event Result
	5	Run marker
	6	Halt marker
	9	Date message
2		Copied data from disk, i.e. previously logged data
	0	All data
3		Messages
	0	Ordinary messages
	7	Power Fail message
	8	Power Back message
4		Listed data: ASCII format, data produced by LIST command
	0	All data
5		Command response: SI3535 replies to commands
	0	Non error responses
	1	Error responses
6		Commands: command data sent to SI3535
	0	All commands
7	0	Operator alert messages
8	0	Result (Standalone data)
	1	Label
	2	Start date
	3	Start Time
	4	Trigger delay
	5	Interval time
	6	Number of results
	7	Maximum value/Minimum value
	8	End of export
	9	Export aborted
9	0	Result (Interactive data)
	1	Label
	2	Start date
	3	Start Time
	4	Trigger delay
	5	Interval time
	6	Number of results
	7	Maximum value/Minimum value
	8	End of export
	9	Export aborted

The data block transmitted, after the establishing link contains a start of header marker, **SOH**, followed by the header characters. The header consists of two ASCII characters which identify the type of data being transmitted in this block. The value and meaning of the header markers used by the SI3535 are given in Table 6.5.

The final character (check character) of the transmitted block is used to check that all characters in the block are received uncorrupted. This block check character (BCC) is 7 bits plus a parity bit (even parity), where each of the first 7 bits are the modulo 2 binary sum of every element in the corresponding bit column, 1 to 7, of successive characters of the transmitted block. The BCC is generated by performing an exclusive OR on all characters transmitted after the **SOH**, up to and including the ETB or ETX character.

Note: The **SOH** characters is not included in the check.

The method of error detection is often referred to as 'the longitudinal parity method', as shown below:


7+1 parity bit
SOH
a b c d e f g P
a b c d e f g P
a etc
a etc
a b c d e f g P
BCC
ETB/ETX
7 bits + even parity
a b c d e f g P
Modulo 2 addition of:
all `a` bits
all `b` bits
all `c` bits

When the data block transmitted was terminated by ETB the line is held so that further data of the same type can be sent when it becomes available. The blocks sent following an ETB must also contain the SOH and Header characters.

Note: The SI3535 will accept concatenated data blocks, terminated by either ETB or ETX. However, it will only transmit discrete blocks, where the first two characters are SOH and Header and the termination character is ETX, This ensures that the line is only held by the SI3535 for the duration of the transmission of a single block of data.

The flow charts in Fig 6.7 and Fig 6.8 show the BS 4505 protocol adopted by the SI3535.

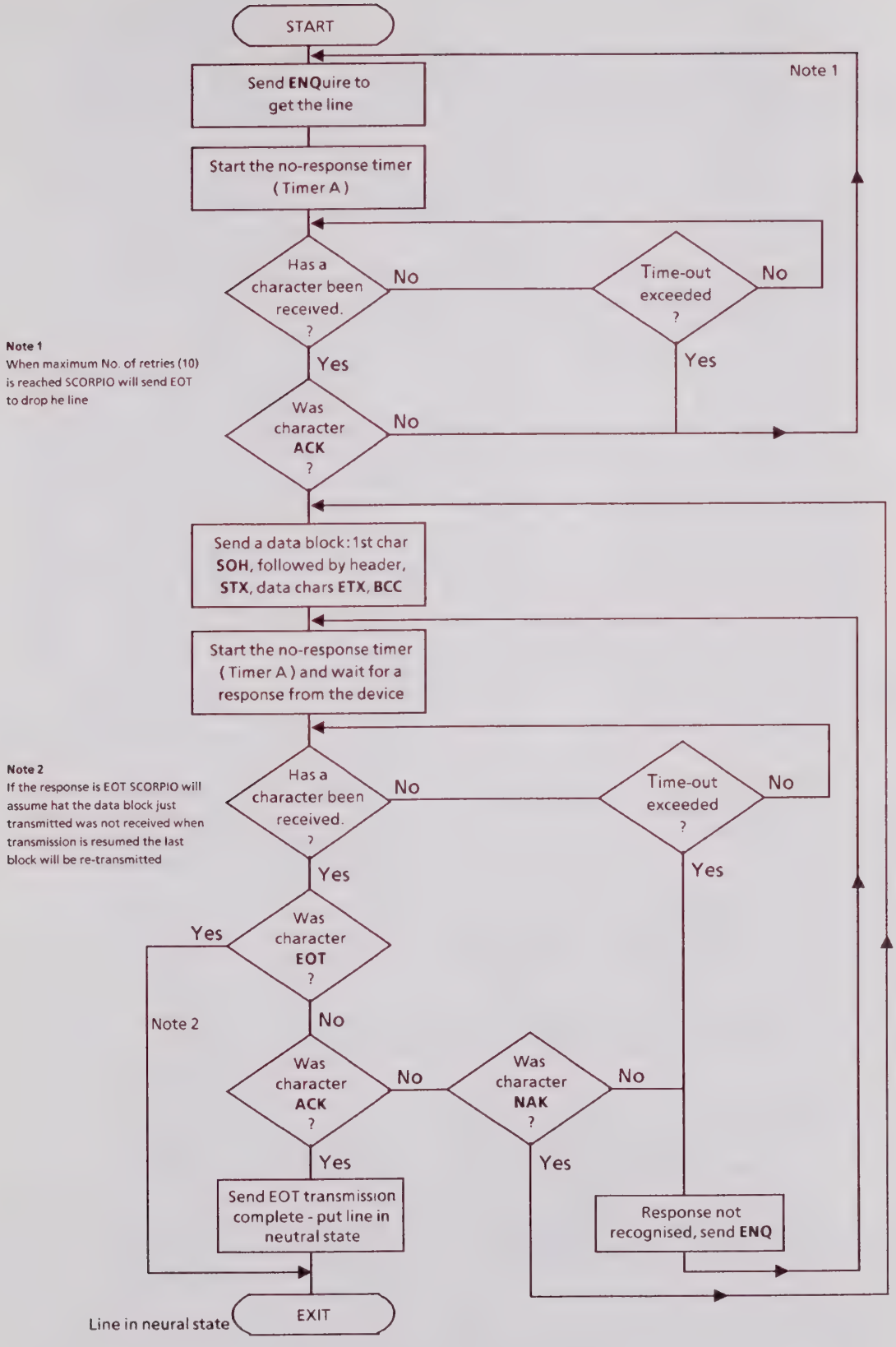

START
Send ENQuire to get the line
Note 1
Start the no-response timer (Timer A)
Has a character been received. ?
No
Time-out exceeded ?
No
Yes
Yes
Was character ACK ?
No
Note 1
When maximum No. of retries (10) is reached SCORPIO will send EOT to drop he line
Send a data block: 1st char SOH, followed by header, STX, data chars ETX, BCC
Start the no-response timer (Timer A) and wait for a response from the device
Has a character been received. ?
No
Time-out exceeded ?
No
Yes
Yes
Was character EOT ?
Yes
Note 2
No
Was character ACK ?
No
Was character NAK ?
No
Yes
Yes
Send EOT transmission complete - put line in neutral state
Response not recognised, send ENQ
Note 2
If the response is EOT SCORPIO will assume hat the data block just transmitted was not received when transmission is resumed the last block will be re-transmitted
Line in neural state
EXIT

Fig 6.7 *BS4505 - Transmit Protocol*

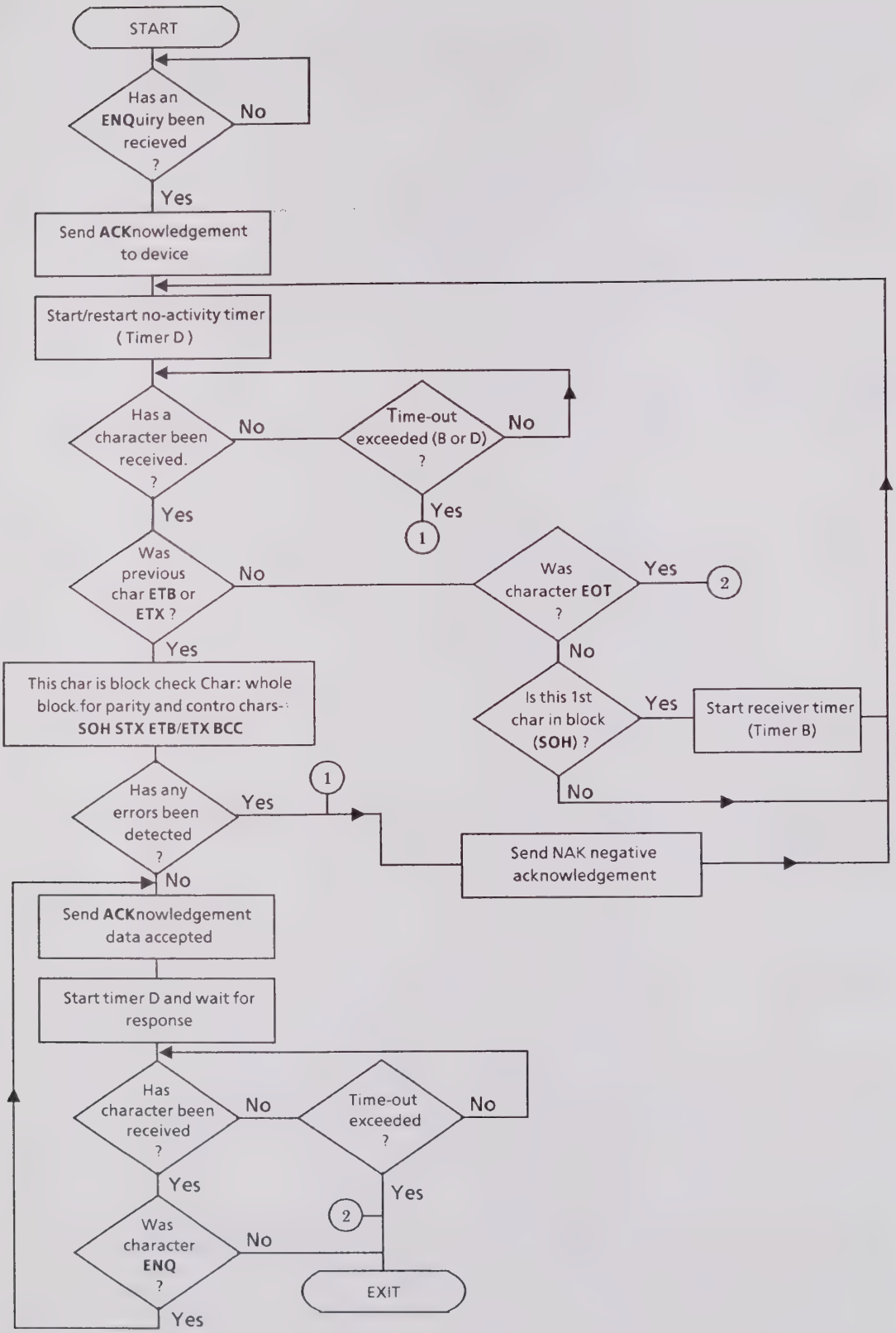


Fig 6.8 *BS4505 - Receive Protocol*

3.3 CONNECTION DETAILS

Information showing pin connection and baud rate settings details are given in Tables 6.6 and 6.7 respectively. The location of the plugs and switches are given in Figures 6.9 & 6.10.

Table 6.6 Serial Interface PL1/PL2 Pin Details

Pin Number	Function	
	RS232	RS422
1	Ground.	Ground.
2	TX output.	-
3	RX input.	RxHi (balanced) input.
4	RTS (Request To Send) output.	-
5*	CTS (Clear To Send)	-
7	0V.	RxLo (balanced) input.
11	-	TxLo (balanced) output.
18	-	TxHi (balanced) output.
20	DTR (Data Terminal Ready) output.	

*CTS line present on PL1 only.

3.3.1 Cable for RS422 Operation

Cables listed in BICC Cableselector No E18 are suitable for use as a connection between equipment in normal environments. For use in more hazardous environments an armoured cable may be required. The EIA standard interconnecting cable characteristics as laid down for RS422 are given below:

a) Conductor size: The interconnecting cable shall be composed of two wires of 24 AWG or larger conductor for solid or stranded copper wires, or for non-copper conductors, a sufficient size to produce a dc wire resistance not to exceed 30 Œ per 1000 feet per conductor.

b) Mutual Pair Capacitance: The capacitance between one wire in the pair to the other wire shall not exceed 20 pF per foot and the value shall be uniform over the length of the cable.

c) Stray Capacitance: The capacitance between one wire in the cable to all others in the cable sheath, with all others connected to ground, shall not exceed 40 pF per foot, and shall be uniform for a given conductor over the length of the cable.

d) Pair-to-Pair Balanced Crosstalk: The balanced crosstalk from one pair of wires to any other pair in the same cable sheath shall have a minimum value of 40 dB attenuation measured at 150 kHz.

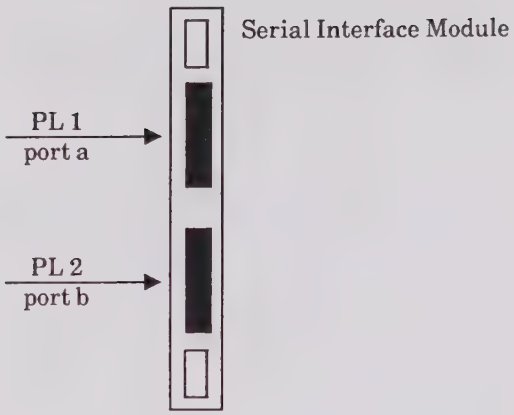


Fig 6.9 *Location of PL1 and PL2 on the Serial Interface Module*

PL1 is output only, with a choice of delimiting characters. PL2 is intended primarily for use by a controlling device but can be used as output only, with a choice of delimiting characters.

Of the two ports, only PL1 appears in the Task Definition menu on the front panel. However, both ports can be programmed either from a remote controlling device or from the front panel.

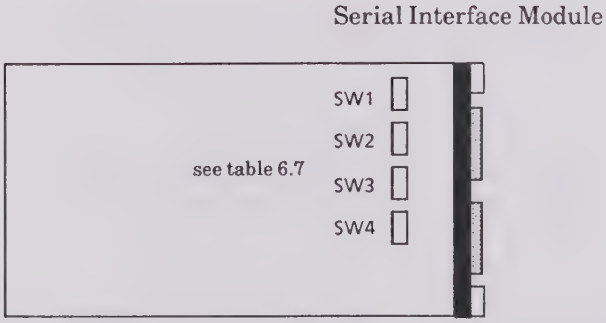
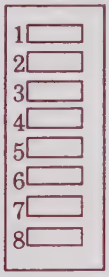


Fig 6.10 *Location of SW1-4 on the Serial Interface Module*

Table 6.7 *Switch Settings for Serial Interface Module*

SW1			
SWITCH CONTACTS	**FUNCTION**	**SWITCH POSITION**	
		OFF	**ON**
1 2 3 4 5 6 7 8	SELECT BAUD RATE FOR PORT A (Continued on SW2)	SET ALL CONTACTS TO 'OFF' EXCEPT THE ONE BAUD RATE SELECTED	153.6kB* 76.8kB* 38.4kB* 19.2kB 9.6kB 4.8kB 2.4kB 1.2kB
SW2			
1	SELECT BAUD RATE FOR PORT a (Continued from SW1)	SET ALL CONTACTS TO 'OFF' EXCEPT THE ONE BAUD RATE SELECTED	600dB
2			300dB
3 4 5 6 7 8	SELECT BAUD RATE FOR PORT b (Continued on SW3)	SET ALL CONTACTS TO 'OFF' EXCEPT THE ONE SELECTED	153.6kB* 76.8kB* 38.4kB* 19.2kB* 9.6kB 4.8kB
SW3			
1 2 3 4	SELECT BAUD RATE FOR PORT b (Continued on SW3)	SET ALL CONTACTS TO 'OFF' EXCEPT THE ONE BAUD RATE SELECTED	2.4kB 1.2kB 600dB 300dB
5	SELECT PARITY FOR PORT a (SW3/6 overrides SW3/5)	EVEN	ODD
6		NO PARITY	PARITY
7	SELECT PARITY FOR PORT b (SW3/8 overrides SW3/7)	EVEN	ODD
8		NO PARITY	PARITY
SW4			
1	SELECT RS232C/422 FOR PORT a	RS422	RS232C
2	SELECT RS232C/422 FOR PORT b	RS422	RS232C
3	SELECT PORT a PROTOCOL	Not Used	
4		C/R, LF & 2 NULLS	C/R
5	SELECT PORT b PROTOCOL	(SW4/6 enabled)	BS 4505 (over -rides SW4/6)
6		C/R, LF & 2 NULLS	C/R

SW1 - 3:
SW4 is similar, but has only 6 switch selections



OFF← → ON

***Note:** When using baud rates of 19.2kB and higher the overall 'throughput' is limited by software execution time rather than baud rate.

4 FRONT PANEL EMULATION

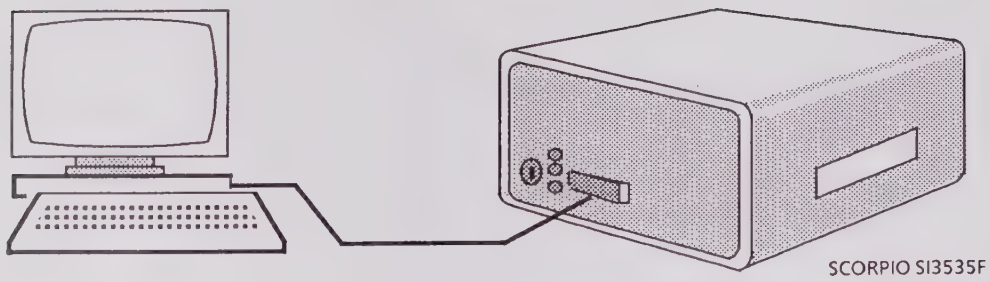


Fig 6.11 *Front Panel Emulator Interconnection*

When connected to the front panel serial port on a SI3535F or G, a VT220 (or compatible), or a PC in terminal mode, can be used to directly emulate the SI3535 front panel.

For further details on front panel emulation refer to the Scorpio User Guide, Chapter 5, Section 2.2.

Chapter 7
GPIB Interface

Section		*Page*
1	**General Information**	7.3
2	**GPIB Status Designations**	7.3
2.1	TALKER ONLY Status	7.3
2.2	TALKER/LISTENER Status	7.4
3	**GPIB Protocol**	7.4
3.1	Bus Management	7.4
3.2	Handshake	7.5
3.3	Data Lines	7.5
4	**GPIB Address**	7.5
5	**Service Requests**	7.6
6	**Polling**	7.8
6.1	Parallel Polling	7.8
6.2	Serial Polling	7.8
7	**Input/Output Data**	7.8
7.1	Input Data	7.8
7.2	Output Data	7.8
8	**Communication via GPIB Bus**	7.8
9	**GPIB Interface Connections**	7.9

Tables

7.1	GPIB Management Lines	7.4
7.2	GPIB Validation Lines	7.5
7.3	GPIB SW1 Switch Settings	7.6
7.4	GPIB Address Settings	7.6
7.5	SI3535 Service Requests	7.7
7.6	GPIB Interface Connections	7.9

Illustrations

7.1	Location of PL1 and SW1 on the GPIB Interface Module	7.5
7.2	GPIB Communication Flowchart	7.9

1 GENERAL INFORMATION

The 35304A module option fully implements the GPIB IEEE 488 (1978) Interface Standard, supporting any GPIB compatible device. Up to 15 instruments can be interconnected by the GPIB Bus, the total length of which must not exceed 20 metres. Devices on the bus are designated as follows:

GPIB Terms

For those unfamiliar with the GPIB, here are the meanings of the more commonly used terms.

- A "Device" is any instrument compatible with GPIB operation.
- A "Talker" is a device that can send information via the GPIB, whilst a "Listener" is a device that can receive this information.
- A "Controller" is a device that is able to assign Talker and Listener roles to itself and to other devices on the GPIB This allows the controller to send control messages to devices on the GPIB and to receive measurement and control status information from them The overall purpose of a controller is to coordinate the operation of all devices on the GPIB, in accordance with a control program.

  Note: Only one device can be designated Controller, a Controller is not absolutely necessary for operation.

The SI3535 is capable of being either a TALKER ONLY, or TALKER/LISTENER device. The TALKER/LISTENER status and address are switch selectable.

The control protocols, cables and connectors required are strictly defined by the GPIB standard to ensure unambiguous inter-device communication. Both parallel and serial polling are implemented.

2 GPIB STATUS DESIGNATIONS

The status designations for the SI3535 when attached to the GPIB BUS are detailed below.

2.1 TALKER ONLY STATUS

When the SI3535 is designated TALKER ONLY status it can be used to output data directly to a Controller, or alternatively to communicate directly with a high speed data recording device such as a printer or tape recorder.

For storage on tape, only the two devices occupy the bus, the data recording device is designated as a LISTENER ONLY device and the SI3535 communicates using Data lines and Validation lines for hardware handshake (refer to Section 3).

2.2 TALKER/LISTENER STATUS

When the SI3535 is attached to a GPIB bus for the purpose of control by a computer it must be designated TALKER/LISTENER status. As a TALKER/LISTENER, the SI3535 can accept commands from another TALKER (eg. computer) and send back data (ie. measurement data, error messages) to that device.

3 GPIB PROTOCOL

The 16 lines comprising the GPIB BUS are defined as follows:

a) Five lines for bus MANAGEMENT (Table 7.1).
b) Three VALIDATION lines - handshake for transfer of data control (Table 7.2).
c) Eight DATA lines - digital input/output (DIO): transfer information or instructions as defined by the MANAGEMENT lines.

3.1 BUS MANAGEMENT

Management lines are used for bus supervision. Signal ATN (Attention) is asserted by the Controller whenever it places an address or command message byte on the bus. ATN is used in conjunction with EOI (End or Identify) when the Controller is carrying out a device poll (the EOI line can also be driven by a TALKER to indicate end of a message).

Table 7.1 *GPIB Management Lines*

LINE	FUNCTION	DESCRIPTION
IFC	Interface Clear	Used by Controller to put bus in known state.
REN	Remote Enable	Used to indicate whether active LISTENER will use programming information sent by user.
ATN	Attention	Indicates whether data line carried control information or data.
EOI	End of Identify	Asserted by TALKER to designate last byte or used in parallel poll.
SRQ	Service Request	Used by device to get attention of Controller.

To alert the Controller that service is required (eg data for transfer), a device asserts SRQ. The Controller then conducts a device poll to determine which device requires service. IFC (Interface Clear) is used by the Controller to initialize the interface to a predetermined state, and REN (Remote Enable) is used to enable an addressed device(s) to accept remote control.

3.2 HANDSHAKE

Message data is transferred under the control of a 3-wire handshake, which runs at the speed of the slowest device in the interchange. A TALKER indicates that a byte is present, by asserting DAV (Data Valid), then waits for all devices to indicate that the data has been accepted. LISTENERS do this by releasing the NDAC (Not Data Accepted) line. Data can only be sent if all devices are ready to receive, indicated by the releasing of the NRFD (Not Ready for Data) line by every device as it becomes ready for another data byte.

Table 7.2 *GPIB Validation Lines*

LINE	FUNCTION	DESCRIPTION
DAV	Data Valid	Asserted by TALKER when Valid data is put on the data line.
NRFD	Not Ready For Data	Held to ground by LISTENER if not ready to receive data.
NDAC	No Data Accepted	Released by active LISTENER when data has been accepted.

3.3 DATA LINES

The data lines are designated DIO1 to DIO8 and can carry either data or service request byte (SRQ) information (DIO1 is the least significant bit).

4 GPIB ADDRESS

The GPIB address enables the Controller to identify each device on the bus, which is designated a unique address in the range 0 to 30. The SI3535 GPIB address is selected by the switch 1 on the GPIB board, SW1/1 to SW1/5 as detailed in Tables 7.3 and 7.4. Figure 7.1 shows the location of the switch.

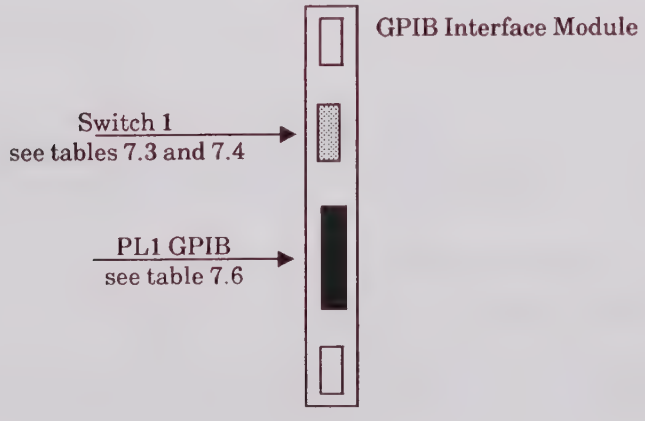


Fig 7.1 *Location of PL1 and SW 1 on the GPIB Interface Module*

Table 7.3 *GPIB SW1 Switch Settings*

Switch Contacts	Function	Switch Setting	
		OFF	ON
1-5	Talker/Listener address	See Table 7.4	
6	Talker/Listener status	Talk only	Talker/listener
7	Not used	-	-
8	Not used	-	-

Table 7.4 *GPIB Address Settings*

Switch Settings (SW1)					Address	Switch Settings (SW1)					Address
/5	/4	/3	/2	/1		/5	/4	/3	/2	/1	
ON	ON	ON	ON	ON	0	OFF	ON	ON	ON	ON	16
ON	ON	ON	ON	OFF	1	OFF	ON	ON	ON	OFF	17
ON	ON	ON	OFF	ON	2	OFF	ON	ON	OFF	ON	18
ON	ON	ON	OFF	OFF	3	OFF	ON	ON	OFF	OFF	19
ON	ON	OFF	ON	ON	4	OFF	ON	OFF	ON	ON	20
ON	ON	OFF	ON	OFF	5	OFF	ON	OFF	ON	OFF	21
ON	ON	OFF	OFF	ON	6	OFF	ON	OFF	OFF	ON	22
ON	ON	OFF	OFF	OFF	7	OFF	ON	OFF	OFF	OFF	23
ON	OFF	ON	ON	ON	8	OFF	OFF	ON	ON	ON	24
ON	OFF	ON	ON	OFF	9	OFF	OFF	ON	ON	OFF	25
ON	OFF	ON	OFF	ON	10	OFF	OFF	ON	OFF	ON	26
ON	OFF	ON	OFF	OFF	11	OFF	OFF	ON	OFF	OFF	27
ON	OFF	OFF	ON	ON	12	OFF	OFF	OFF	ON	ON	28
ON	OFF	OFF	ON	OFF	13	OFF	OFF	OFF	ON	OFF	29
ON	OFF	OFF	OFF	ON	14	OFF	OFF	OFF	OFF	ON	30
ON	OFF	OFF	OFF	OFF	15	OFF	OFF	OFF	OFF	OFF	31

When selecting an address for a SI3535 unit, care should be taken to ensure that the address is not the same as that of any other device on the bus.

5 SERVICE REQUESTS

Before the SI3535 can to send data to another device on the bus it must first gain the attention of the Controller by sending a service request. This is achieved by setting the Service Request (SRQ) line. In addition, the SI3535 will set up status information for the data lines DIO1 to DIO8 as follows:

DIO7 Set Request for service bit (RQS)

DIO1 to DIO6 Service Request type data.

In response to the request for service the Controller can perform either a serial poll or parallel poll of the SI3535 and the other devices on the bus.

A parallel poll would generally be used where there is more than one device on the bus that may require service, in order to establish which device has issued the request. A serial poll may be used by single or multi-device systems, to interrogate the data byte for status information on the requesting device (eg the SI3535).

Table 7.5 lists the SI3535 service request types together with the value of the resultant data byte as shown below. Bits 0 to 5 of the byte contain the service request type, bit 6 is always set (RQS bit) and bit 7 is not used, so it is always zero.

Table 7.5 *SI3535 Service Requests*

REASON FOR REQUEST	VALUE OF DATA BYTE		
	BINARY	DECIMAL	HEX
Monitor Data available	01000000	64	40
Logged and/or processed data available	01000010	66	42
Data copied from the 3535 disk available	01000100	68	44
Message available, e.g. Power Fail/back, "WRITE" data, run time error message, etc.	01000110	70	46
Listed data available (LIST command)	01001000	72	48
Command response data available (query data, Error response, Immediate measurement results.)	01001010	74	4A

NOTE: All other values of service request are invalid

6 POLLING

6.1 PARALLEL POLLING

The PPI subset is followed in parallel polling. Before the SI3535 responds to a parallel poll command, the response bit configuration for a specific unit must be set up remotely by the Controller. This must be done using the parallel poll configuration command applicable to the Controller being used.

When parallel-polled by the Controller, if requiring service, the SI3535 responds by setting its designated response bit in the data byte. The Controller can then establish the link for the SI3535 to send its data. However, before establishing the link the controller may serially poll the SI3535 to see what type of service request has been made.

6.2 SERIAL POLLING

When the Controller polls the SI3535 serially, in response to a service request or parallel poll, it interrogates the status information set up by the SI3535 in the data byte, DIO1 to DIO8. If the SI3535 has sent a Service Request (SRQ) the Request for Service bit, bit 6, will have been set, and bits 0 to 5 will contain the type of service request. The Controller may now establish the link for the SI3535 to send data.

7 INPUT/OUTPUT DATA

7.1 INPUT DATA

Input data command strings sent to the SI3535 can be up to 80 characters in length including terminating characters. The terminating characters may be either C/R LF or C/R only with EOI asserted on the final character.

Note: If the data string is 80 characters long, EOI need not be asserted. If the data string is less than 80 charcters long, EOI must be asserted on the final character.

7.2 OUTPUT DATA

Output from the SI3535 is in 20, 40,60 or 80 character blocks, terminated by linefeed, (LF) with EOI asserted. The LF is additional to the 20, 40,60 or 80 characters in the block.

8 COMMUNICATION VIA GPIB BUS

Communication via the GPIB Bus is outlined in Figure 7.2. The example shows a routine to read data from and send commands to a the SI3535 unit via the GPIB bus.

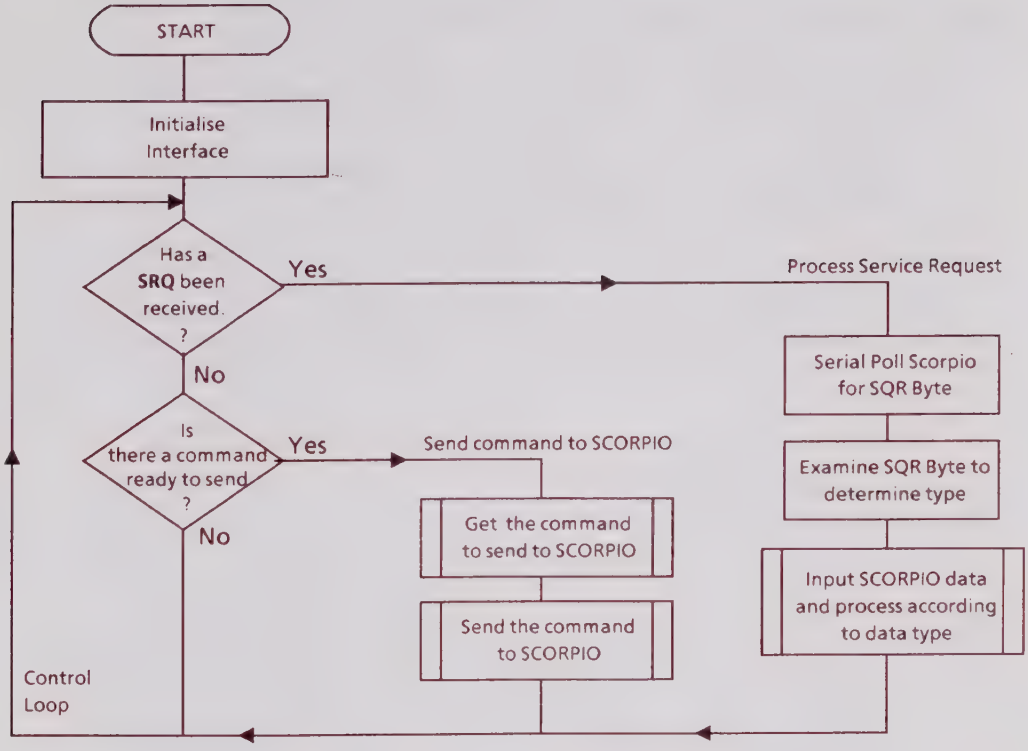


Fig 7.2 *GPIB Communication Flow Chart*

9 GPIB INTERFACE CONNECTIONS

The GPIB Bus is a standard cable, containing 16 signal line wires and eight earth returns, to link devices in the system. Cable connectors comprise a plug/socket "piggy-back" arrangement, allowing additional cables to be connected, maintaining the parallel bus structure Pin connections and switch settings are given in Table 7.6.

Table 7.6 *GPIB PL1 Pin Details*

Pin No.	Signal	Pin No.	Signal	Pin No.	Signal	Pin No.	Signal
1	D101	7	NRFD	13	D105	19	0V
2	D102	8	NDAC	14	D106	20	0V
3	D103	9	IFC	15	D107	21	0V
4	D104	10	SRQ	16	D108	22	0V
5	EOI	11	ATN	17	REN	23	0V
6	DAV	12	-	18	0V	24	0V

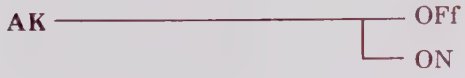
Appendix 1
Command Summary

The SI3535 Command Language can be used either for front panel control of a SI3535D; local terminal control of SI3535F and G, or remote control of all three versions.

Related SI3535 commands are grouped together, to give a tree-like structure referred to as a Command Tree. The command trees define all the possible options associated with a specific function (eg. definition of an event task, single channel measurements) Many of the commands shown in a specific tree are optional, and the users only need to enter the options required.

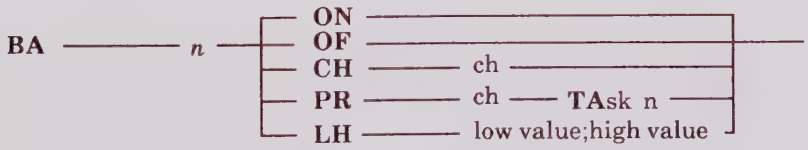
ACKNOWLEDGE COMMAND

COMMAND TREE STRUCTURE:



BARGRAPH COMMAND

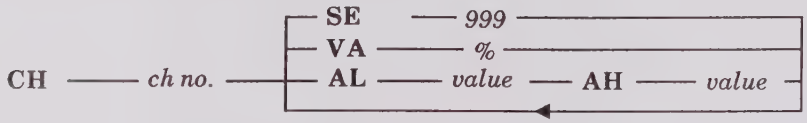
COMMAND TREE STRUCTURE :



CHANNEL DEFINITION :

ANALOG OUTPUT CHANNEL

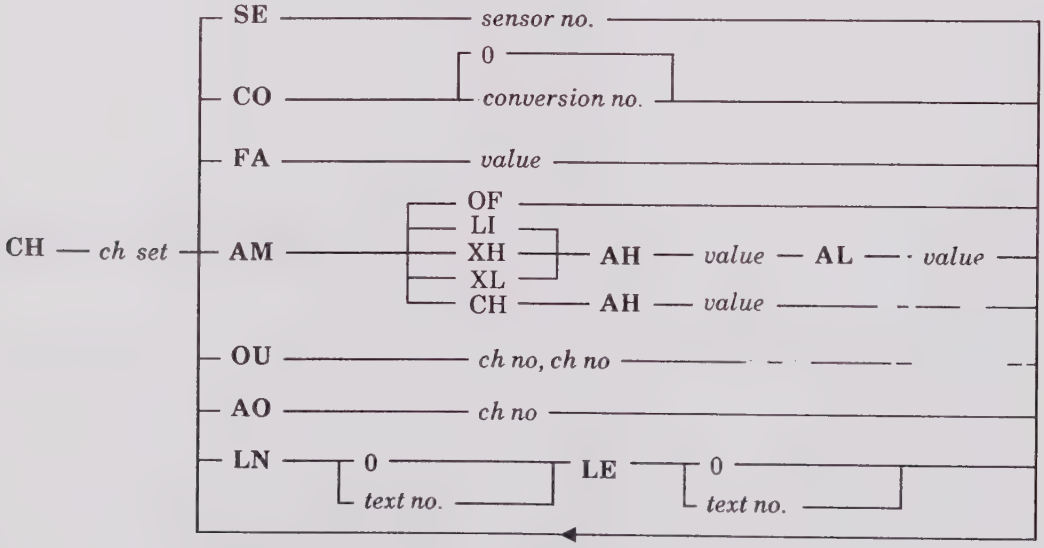
COMMAND TREE STRUCTURE:



CHANNEL DEFINITION :

ANALOG INPUT CHANNEL

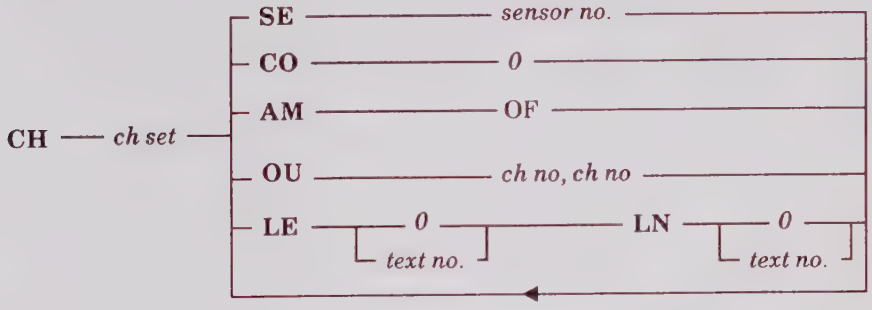
COMMAND TREE STRUCTURE:



CHANNEL DEFINITION :

COUNTER-TIMER DIGITAL INPUT CHANNEL

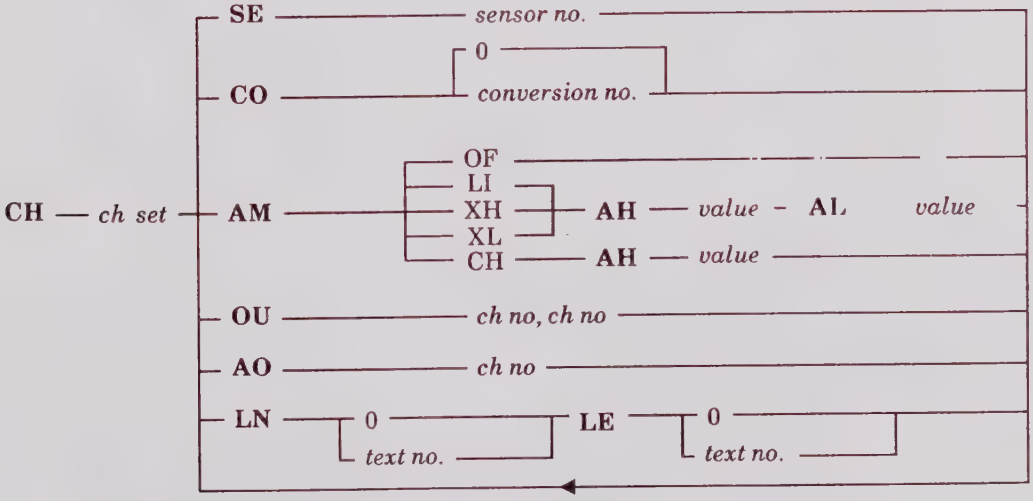
COMMAND TREE STRUCTURE :



CHANNEL DEFINITION

DECIMAL - TYPE DIGITAL INPUT CHANNEL

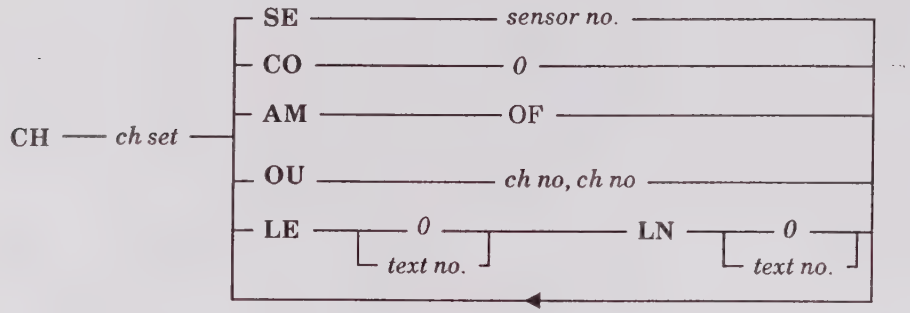
COMMAND TREE STRUCTURE:



CHANNEL DEFINITION :

EVENT - TYPE DIGITAL INPUT CHANNEL

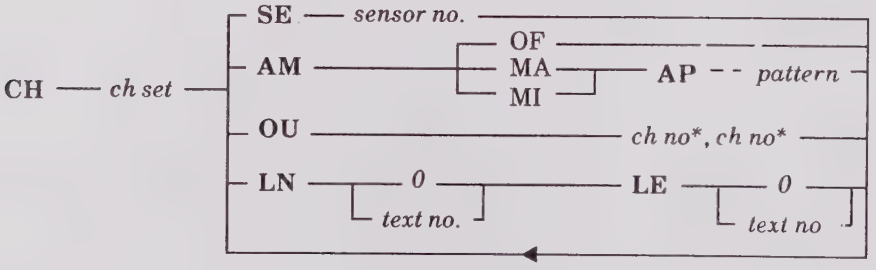
COMMAND TREE STRUCTURE :



CHANNEL DEFINITION :

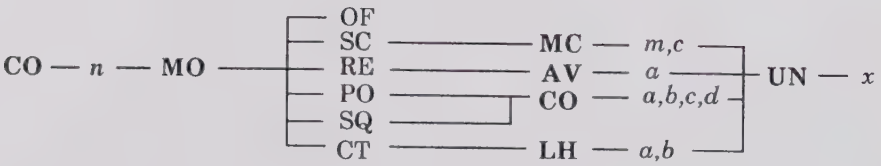
STATUS - TYPE DIGITAL INPUT CHANNEL

COMMAND TREE STRUCTURE:



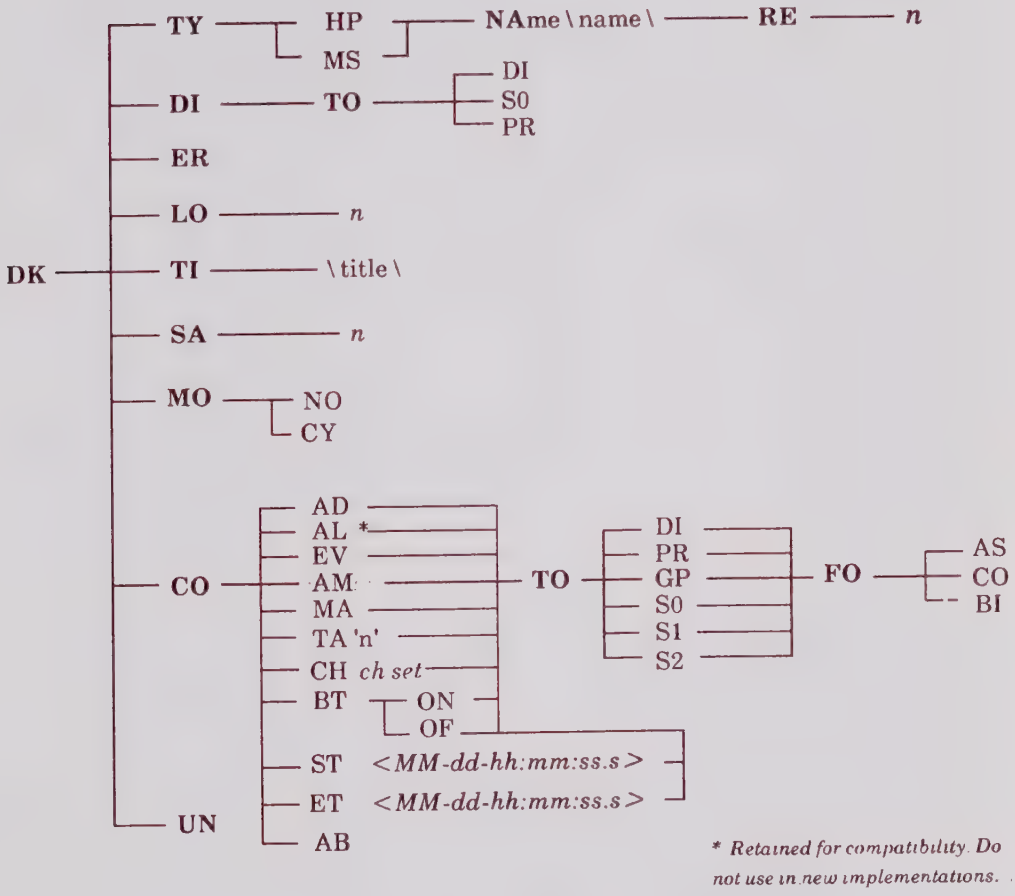
CONVERSION DEFINITION COMMANDS

COMMAND TREE STRUCTURE :



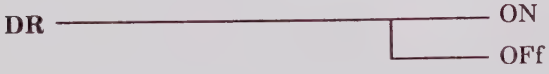
DISK COMMAND

COMMAND TREE STRUCTURE:



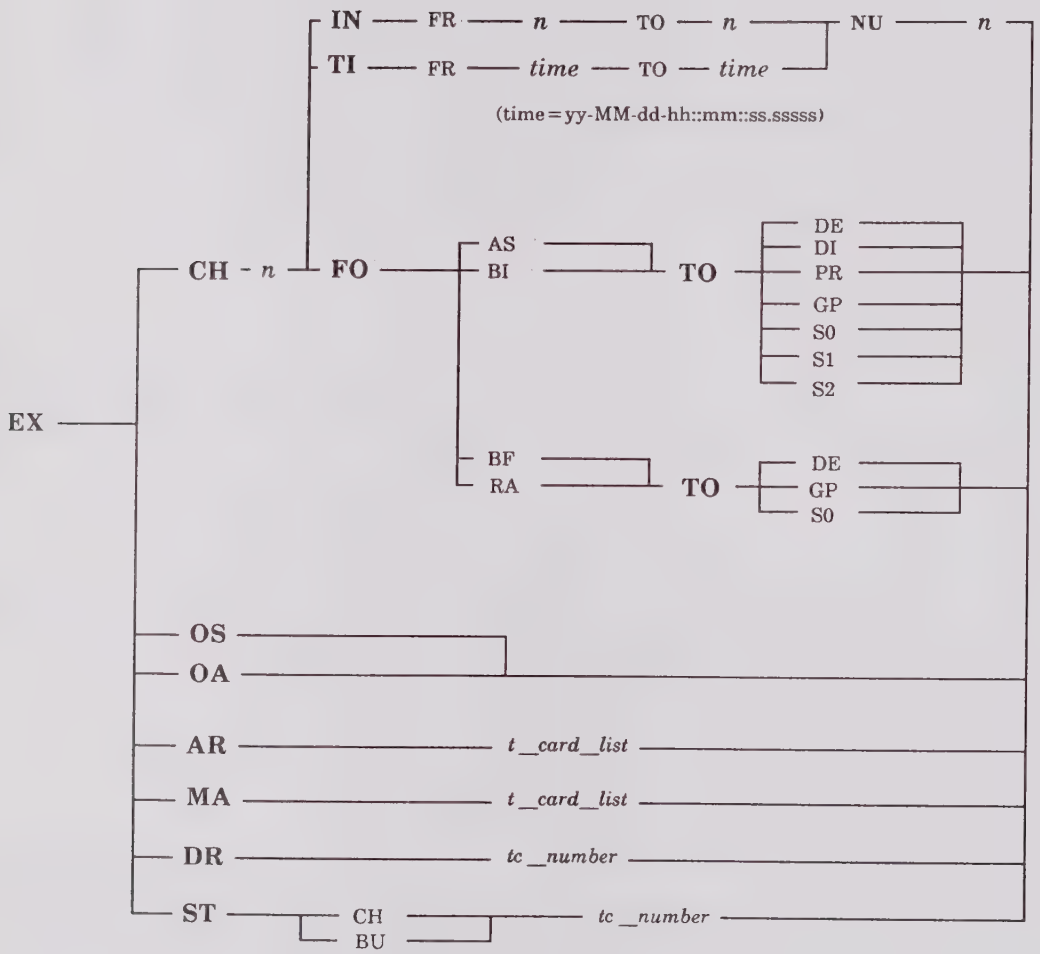
DRIFT COMMAND

COMMAND TREE STRUCTURE :



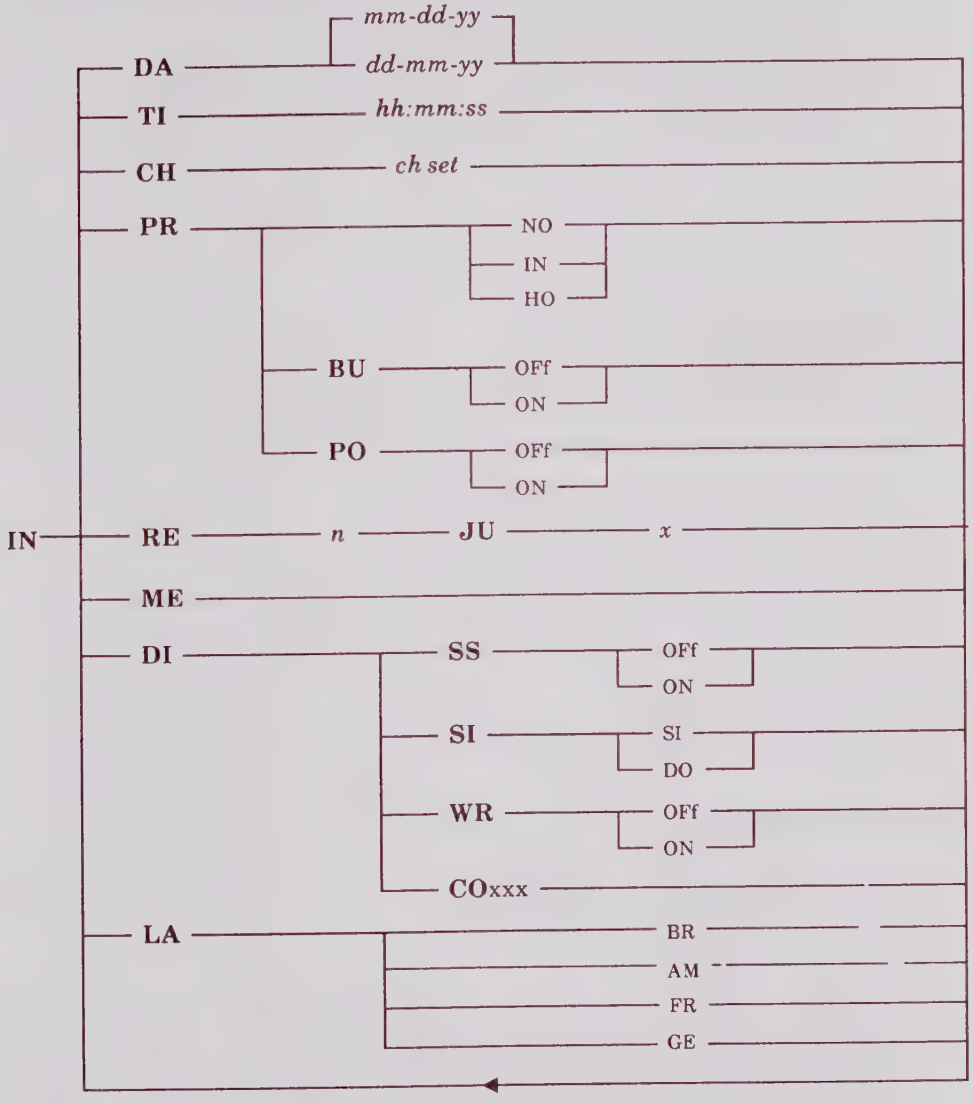
EXPORT COMMAND

COMMAND TREE STRUCTURE :



INITIALIZATION COMMAND

COMMAND TREE STRUCTURE:

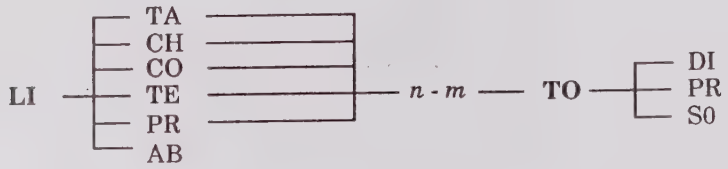


TEXT DEFINITION COMMAND

COMMAND TREE STRUCTURE :

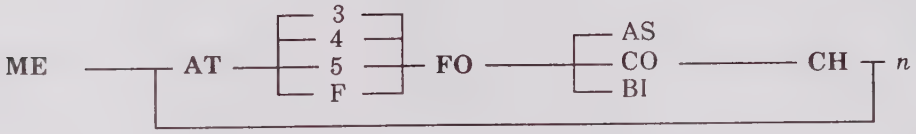
LIST COMMAND

COMMAND TREE STRUCTURE :



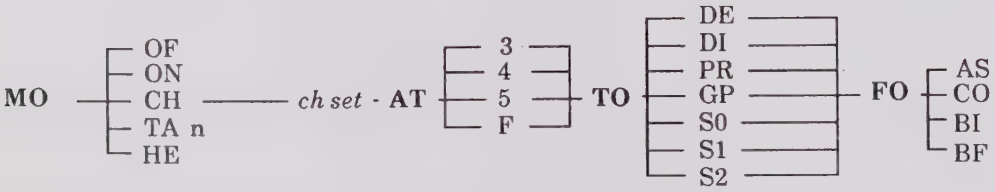
MEASURE COMMAND

COMMAND TREE STRUCTURE :



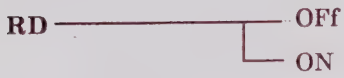
MONITOR COMMAND

COMMAND TREE STRUCTURE :



DIGITAL IO CARD RESET ENABLE/DISABLE COMMAND

COMMAND TREE STRUCTURE:

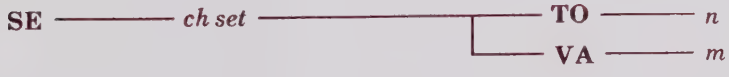


RENUMBER COMMAND

COMMAND TREE STRUCTURE :

SET COMMAND

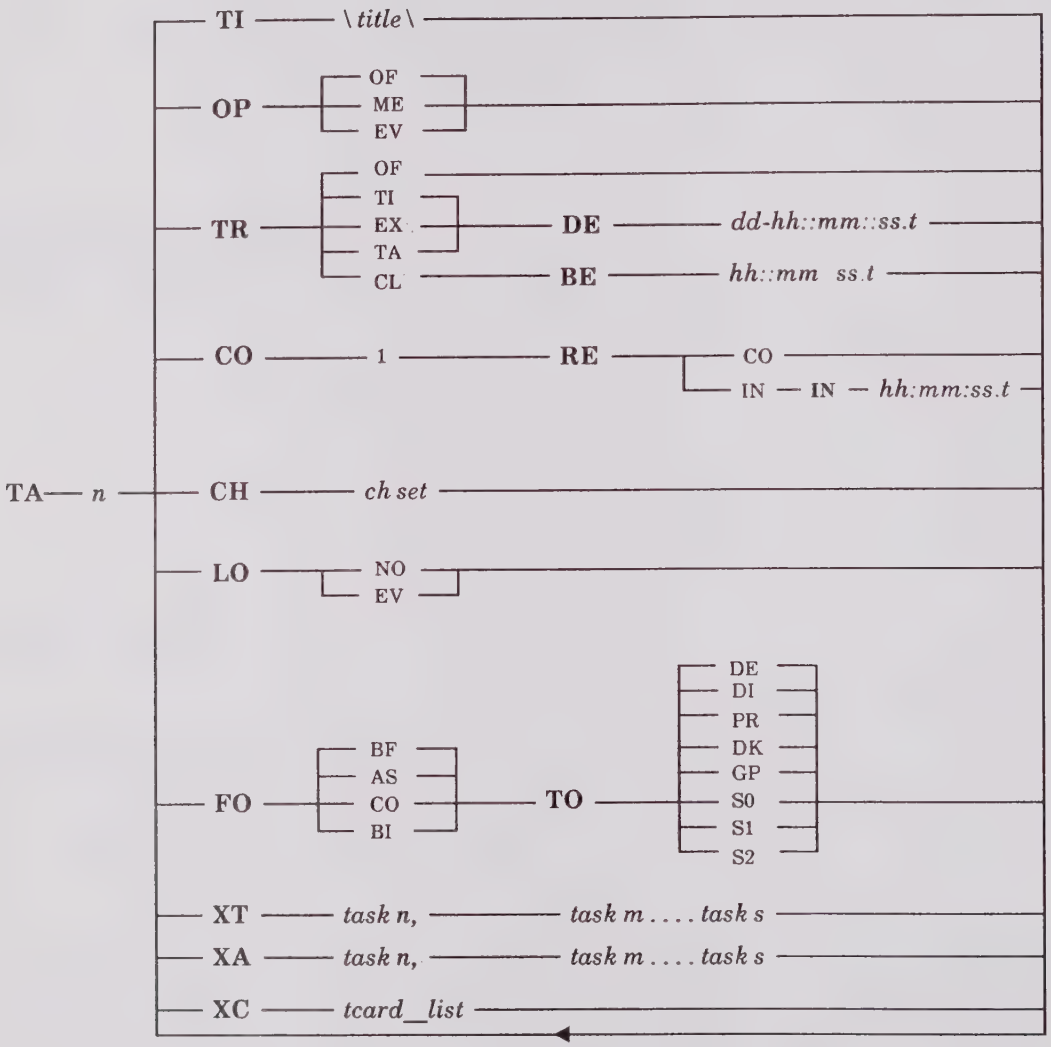
COMMAND TREE STRUCTURE :



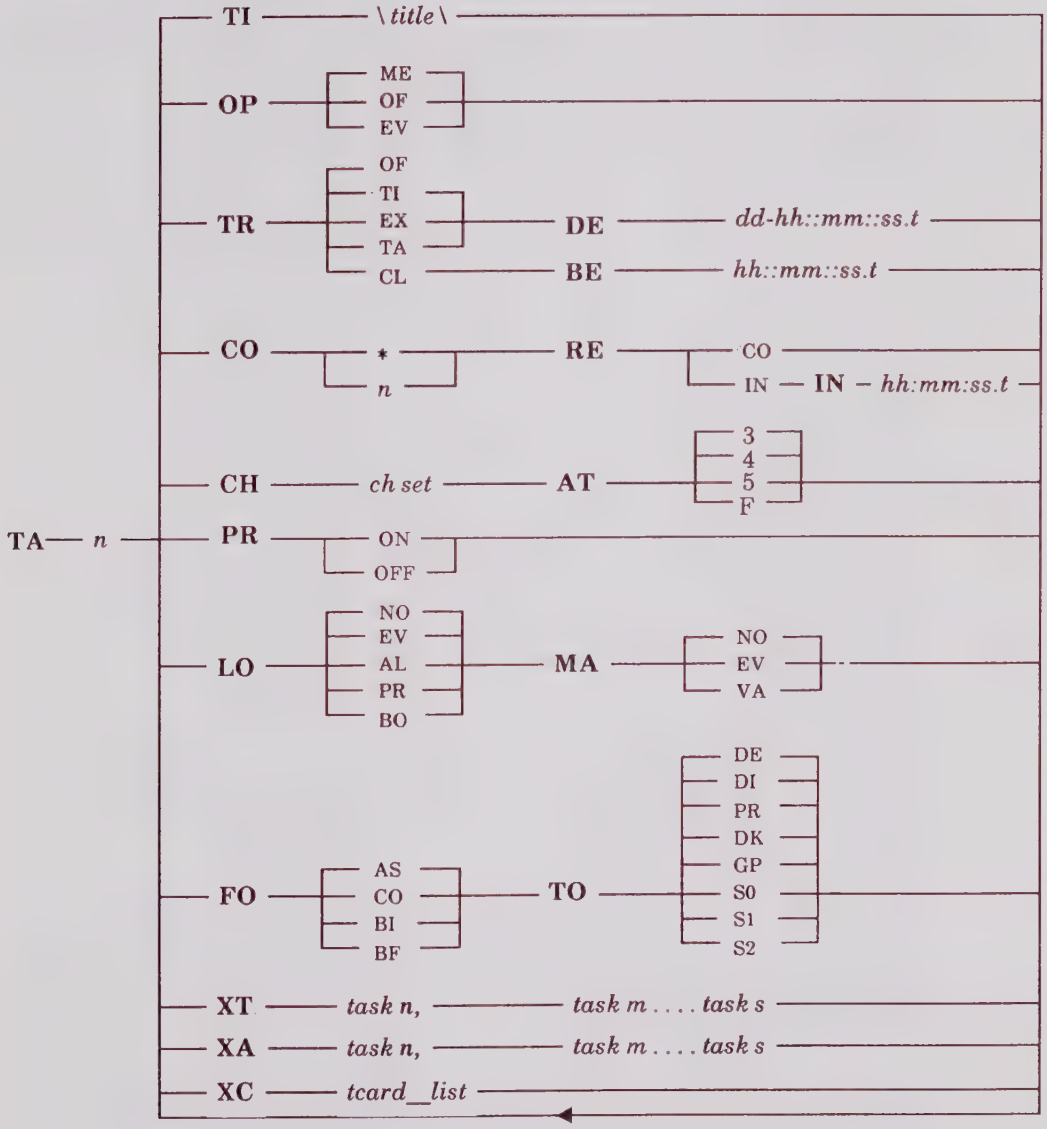
TASK DEFINITION :

EVENT TASK

COMMAND TREE STRUCTURE :

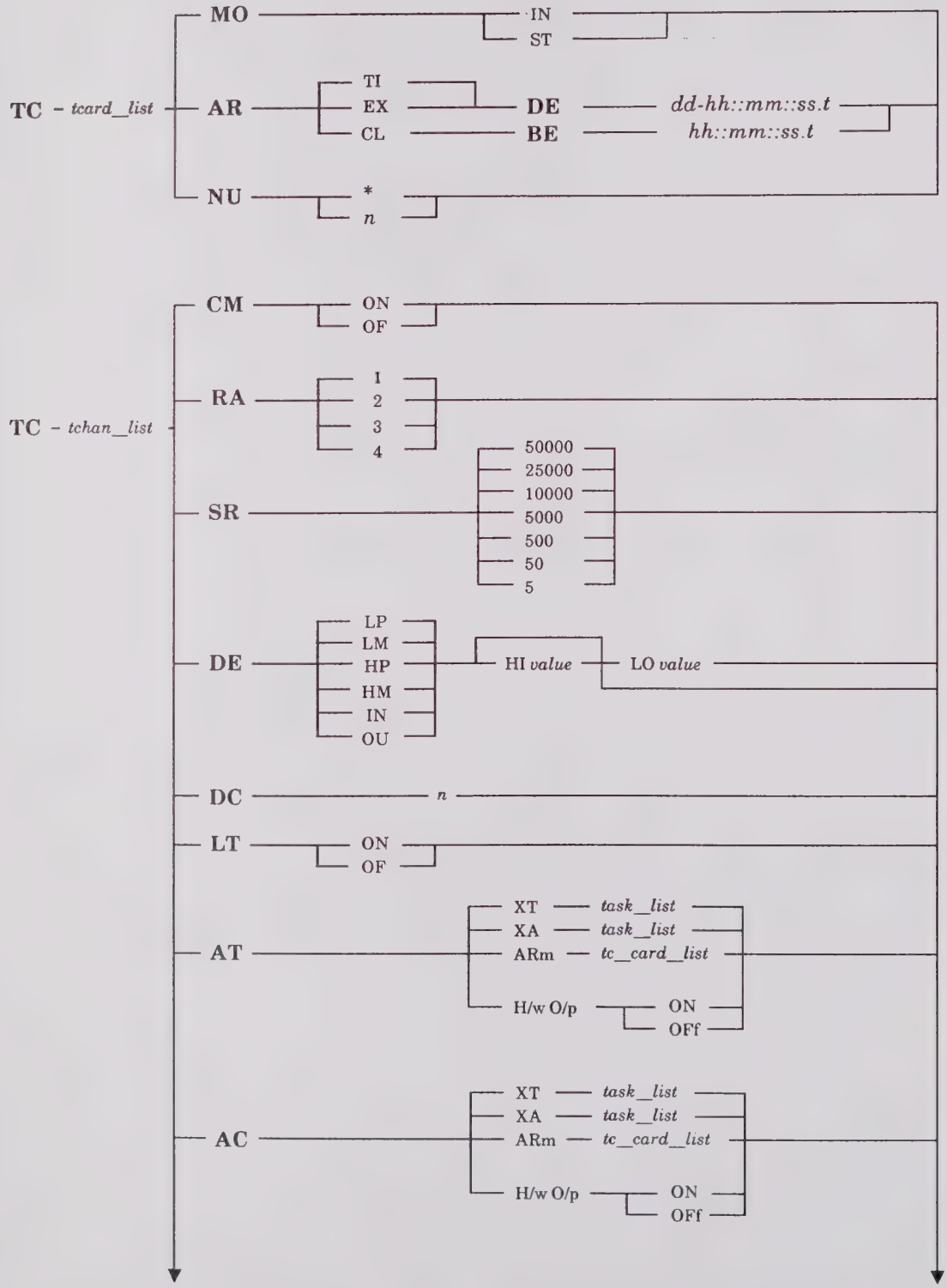


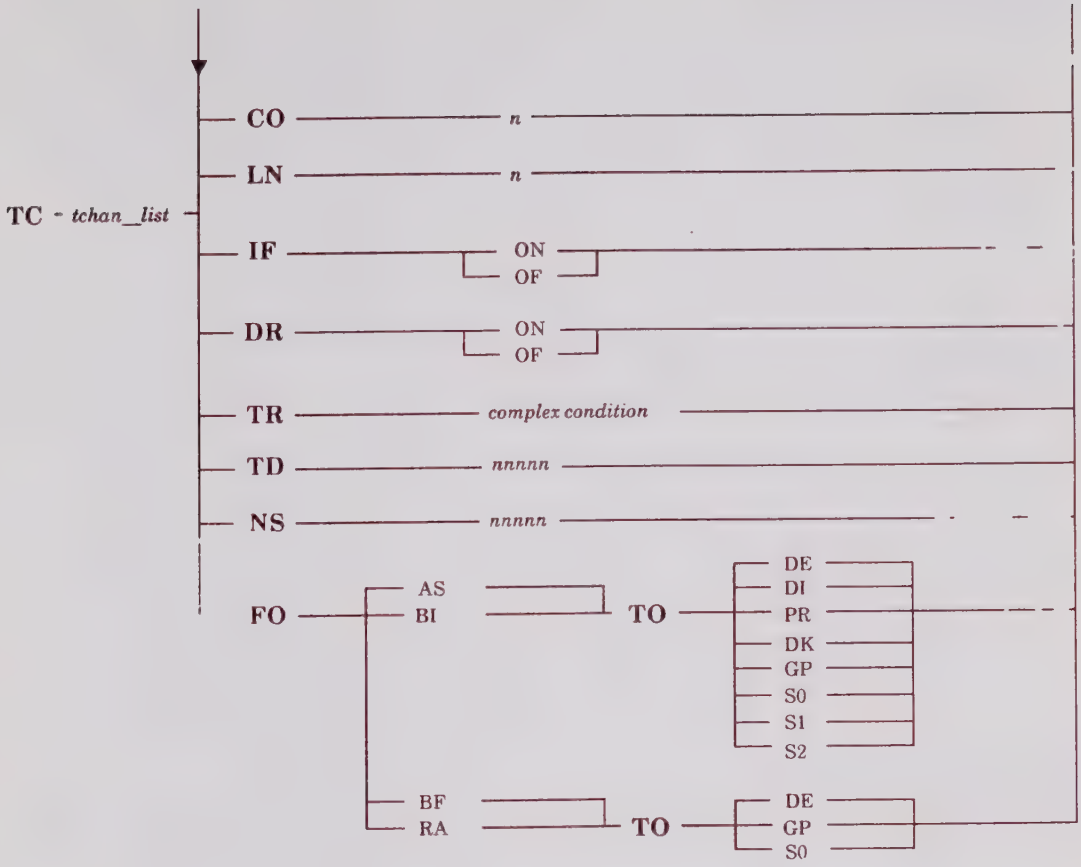
COMMAND TREE STRUCTURE:



TRANSIENT CAPTURE

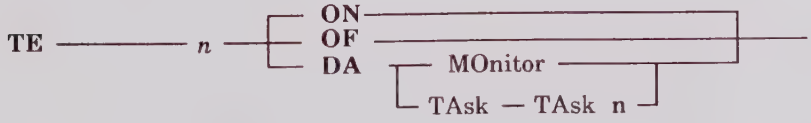
COMMAND TREE STRUCTURE :





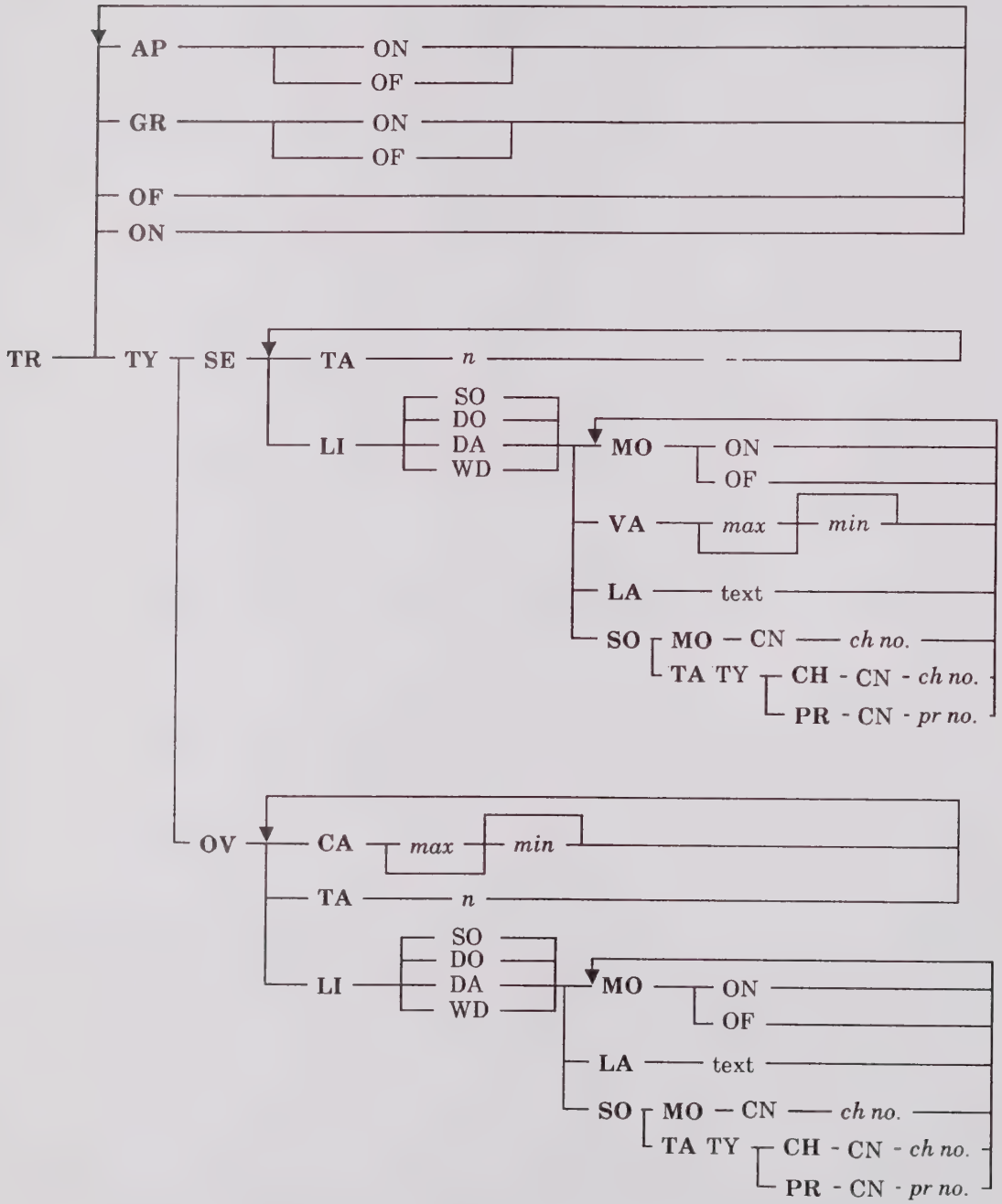
TEMPLATE COMMAND

COMMAND TREE STRUCTURE :



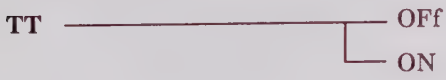
TRENDS COMMAND

COMMAND TREE STRUCTURE:



TIME TAG COMMAND

COMMAND TREE STRUCTURE:



WRITE MESSAGE COMMAND

COMMAND TREE STRUCTURE :

CONTROL COMMANDS

RU

HA

PA

GO

XT

XA

PR SC

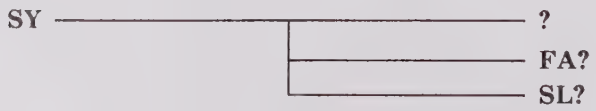
QUERY COMMANDS

Interface Type Command Tree Structure :

Digital Status Command Tree Structure :

Analogue Output Module Configuration Command Tree Structure :

Clock Synchronization Command Tree Structure :



Appendix 2
Sensor Number Tables

The sensor number codes for Event Status and Digital I/O, Analogue Selectors and Counter Timer are given in Tables A2.1, A2.2 and A2.3 respectively.

Table A2.1 *Sensor Numbers for Event Status and Digital I/O*

1st DIGIT		2nd DIGIT		3rd DIGIT		COMMENTS
7	EVENT STATUS CARD	0	BIT pattern input	1-8	Number of bits grouped	For individual or grouped status inputs, a number of digital levels can be examined by a scanning task (eg alarm condition)
		1	BCD	1-5	Number of decades	For BCD inputs, each BCD decade uses 4 consecutive input channels, these are converted by the 3535 into a decimal number.
		2	Inverted BCD	1-5	Number of decades	For inverted (negative logic) BCD inputs.
		4	Totalise	0 1 2	Hi-Lo Lo-Hi Both	Cumulative total of input pulses (transitions) Maximum total - 65535
		5	Count	0 1 2	Hi-Lo Lo-Hi Both	Count of input pulses for scan. Count reset at start of each scan.
		6	Event capture	0 1 2	Hi-Lo Lo-Hi Both	Channels continuously monitored for events. All events detected are regarded as alarms
		7-9	Not used			
8	DIGITAL I/O CARD	0	BIT pattern input	1-8	Number of bits grouped	For individual or grouped status inputs. Inputs and outputs are used independently. Inputs provide status information, outputs give alarm or control (stimulus) information.
		1	BCD	1-6	Number of decades	
		2	Inverted BCD	1-6	Number of decades	External module measurement (port mode): can be used to read binary or BCD data from external modules.
		3	Binary	1-3	Number of 8 bit bytes	

Table A2.2 *Sensor Numbers for Analogue Selectors*

1st DIGIT	2nd DIGIT	3rd DIGIT	COMMENTS
1 VOLTS	1 DC	0 Auto range 1 10mV 2 100mV Range 3 1V 4 10V 5 100V	
	2 AC	0 Auto range 1 20mV 2 200mV Range 3 2V 4 20V 5 200V	
2 RESISTANCE	4-terminal 1 1 mA 2 100µA 3 User defined 2-terminal 5 1 mA 6 100µA 7 User defined	0 Auto range DC voltage Range 1 10mV without thermal 2 100mV drift 3 1V 4 10V 5 Auto range Range with 6 10mV thermal drift 7 100mV compensation 8 1V 9 10V	Result will be in Ω, except when current is user defined. In this case a 100µA current is assumed and conversion must be applied to the result.
3 THERMOCOUPLE	1 Type E 2 Type J 3 Type K 4 Type R 5 Type S 6 Type T 7 Type B 8 Type N	0 Ambient 1 Ext Ref 1 Thermocouple 2 Ext Ref 2 reference 3 Ext Ref 3 junction 4 Ext Ref 4 compensation 5 Ext 0°C	
4 PLATINUM RESISTANCE THERMOMETER	4-terminal 1 1 mA 2 100µA 3 User defined 2-terminal 5 1 mA 6 100µA 7 User defined	0 Auto range Range without 1 10mV thermal drift 2 100mV compensation 3 1V 4 10V 5 Auto range Range with 6 10mV thermal drift 7 100mV compensation 8 1V 9 10V	
5 CURRENT	1 DC	0 Auto range Range assuming 1 100µA 100Ω shunt 2 1mA 3 10mA 4 10mA	

Table A2.2 *Sensor Numbers for Analogue Selectors [continued]*

1st DIGIT	2nd DIGIT	3rd DIGIT	COMMENTS
5 CURRENT	2 AC	0 Auto range 1 200µA 2 2mA 3 20mA 4 20mA	Range assuming 100Ω shunt
6 STRAIN GAUGE	1/2 or 1/4 bridge (dual current) 1 1 mA 2 100µA 3 User defined full bridge 4 8.33mA 5 1 mA 6 100µA 7 User defined 8 2.0V 9 10V	0 1/2 bridge 1 1/4 bridge 120Ω 2 1/4 bridge x 3 1/4 bridge y 4 1/4 bridge z 5 1/2 bridge 6 1/4 bridge 120Ω 7 1/4 bridge x 8 1/4 bridge y 9 1/4 bridge z 0 full bridge 5 full bridge	Range without thermaldrift compensation With thermal drift compensation No thermal drift compensation Thermal drift compensation

Table A2.3 *Sensor Numbers for Counter Timer Module*

1st DIGIT	2nd DIGIT	3rd DIGIT	COMMENTS
9 COUNTER TIMER MODULE	0 Frequency	0 External gate 1 1 second Interval gate 2 1 second " " 3 10 seconds " " 4 100 seconds " "	
	1 Totalize	0 External gate	
	2 Period	1 1 2 10 3 100 4 1000 5 10,000	Numbers of periods averaged
	3 Interval	0 Interval gate	
	4 Off	0 Counter time function off	
0 COUNTER TIMER MODULE	4 Off	0 Counter time function off	May be used instead of 940 to set counter timer function off

Sensor Number for Analog Output Module

An analog output channel must have the sensor number 999 in order that it may be 'scanned' or 'monitored'.

Appendix 3
Error Messages

The SI3535 provides an automatic indication of detected errors. Programming errors and other faults, summarized below, are indicated by error messages on the front panel display or VDU, whereas measurement errors are included in the output results.

ERROR No.	ERROR	REMEDY
000	No error	Used as acknowledgement.
001	Unrecognized command	Re-enter, starting at the beginning of the function.
003	No query available	The requested information is not available.
021	Illegal numeric input	Re-enter numbers correctly.
031	Illegal command	Re-enter, starting at the beginning of the function.
041	Illegal option	Re-enter, starting at the beginning of the function.
061	Failed to initialize strain gauge	Check for damaged, or over- not possible strained gauge. Check for broken connection.
088	End of data-overrun	(See note 1)
089	End of data	
090	Disk full	
091	No disk in drive	
092	Data write protected	Select 'Write' on disk, or to avoid overwriting on this disk fit another set to 'Write'.
093	Disk changed	Disk must not be changed while the SI3535 is in RUN mode - unless paused. (See DisK UNload)
094	Program not found	A space has not been reserved for the program or no program has been saved to that number.
095	Disk drive fault	Attempt to use Disk Drive again. Call the Service Engineer.

Note 1: *If data being logged to the disk in Cyclic mode should 'catch-up' with and attempt to overwrite the data being currently replayed from the disk, then the error message "end of data - overun" will be generated and replay will stop. Logging of data will continue.*

ERROR No.	ERROR	REMEDY
096	Read failed	Try procedure again. If error persists, replace disk.
097	Write failed	Try procedure again. If error persists, replace disk.
098	Bad format	Cannot be read by the SI3535. Try to read on HP or IBM computer.
099	Disk busy	When repalying, wait until the disk has stopped being used.
100	Disk not fitted	
110	Not in local control	Press LOCAL and wait for local control to be granted, or use remote control device to reliquish control of the SI3535.
113	Local lockout active	Disable local lock-out condition on remote controller.
118	Logger running	Program cannot be changed while logger is set to RUN, select HAlt is necessary.
119	Monitor active	Program cannot be changed while logger is Monitoring; de-select Monitor if necessary.
127	BASIC syntax error	
150	Illegal line number	
151	No room for program	
152	No room for variable	
153	Illegal element in program	Re-read the relevant section in Chapter 4 to ascertain correct form.
154	FOR/NEXT nesting error	"
155	IF/ENDIF nesting error	"
156	BASIC syntax error	
157	Line too long	
158	Parenthesis depth error	
159	Illegal variable name	Re-read the relevant section in Chapter 4 to ascertain correct form.

ERROR No.	ERROR	REMEDY
160	Variable name too long	
161	Renumber failed	
162	No such line	Check line number. Errors may be part of a GOTO instruction.
163	Outside current program	
164	No room on line for token	
165	No more lines available in template	
171	Program corrupted	The SI3535 requires servicing.
172	Illegal operand	
173	Index error (eg. (I) = -, or -ve or > 600)	Replace with a permissible index number.
174	Run Time: channel not present	Either correct the channel number in the program, or add the channel to the Task Definition.
175	Formula too complex (ie. more than 4 pending operations.)	Split the formula into two separate lines of program.
180	Transient Output busy	
181	Not transient channel	
182	Inappropriate output device	
183	Bad complex trigger	
184	Insufficient memory	
185	Trigger delay out of range	
186	Disk in cyclic mode	
187	Operation only allowed for interactive channel	
188	Channel not disarmed	
189	Channel has not been armed	

ERROR No.	ERROR	REMEDY
191	No buffer status available	
192	No data available	
193	Number of results too small	
195	Trend graph active	
200	Template running	
201	Channel marker badly formed	
202	Line expands past 80 characters	
203	Too many lines in template	
204	Line numbers not in same template	
205	Monitor source	
210	Test mode switch off	
211	Incorrect cards fitted	
212	Input voltages outside calibration limits	
213	Calibration failure	
252	Inappropriate TEM	
253	No energisation module	A Transducer Energisation Module (35351D) must be fitted in order to measure the sensor type specified.
254	Channel number conflict	The channel numbers specified do not correspond (wholly or partially) with the type or quantity of Input Selectors fitted. Enter the correct channel numbers.
255	Function not applicable (eg. alarms not applicable to event channels)	The requested function is not available.
256	Error	

Disk Handling Messages

When in RUN mode or when replaying data from the SI3535 disk, one of the following messages may be output to the controlling device.

MESSAGES	COMMENT
*disk full	Results file is full.
*no disk in drive	
*write protected	
*disk changed	Disk was changed during RUN.
*program not found	
*drive fault	Hardware fault.
*read failed	Read operation failed
*write failed	Write operation failed
*bad format	Unrecognized format (not MSDOS or HP-LIF)
*disk busy	Data being replayed (copied) from disk
*end of data-overrun	(See Note 1)

Note 1: *If data being logged to the disk in Cyclic mode should 'catch-up' with and attempt to overwrite the data being currently replayed from the disk, then the error message "end of data - overun" will be generated and replay will stop. Logging of data will continue.*

Measurement Errors Included in the Output Result

If an error is detected, one of the following 9 error messages will be output instead of the numerical value of the result.

ERROR MESSAGES	ERROR INDICATED
E1 Ovrld	Measurement overload
E2 Ovrld	Measurement out of linearisation range
E3 Ambnt	Invalid ambient temperature
E4 Ovrfl	Event total exceeds 65536
E5 TmOut	Timeout in external module measurement (ie >20s)
E6 Initl	Strain gauge channel not initialized
E7 NoConn	Input connector not fitted
E8 Wait	Result not yet accumulated (Counter Timer Module)
E9 Off	Counter Timer Channel is off

Error messages are included in Full format and Compact format results.

SI3535 REFERENCE MANUAL INDEX

A

Acknowledge 3.8
Analog channels 3.6
Analog input channel 3.11
Analog output channel 3.14
Analog output module configuration 3.61
Arm 3.64, 3.72.
ASCII 3.4
At Complete 3.66
At Trigger 3.66
Auto ranging 3.6

B

Bargraph 3.42
BASIC 1.5, 4.3 – 4.4
Binary format 3.33, 3.38, 5.3, 5.11, 5.25
BS4505 1.4, 6.16
BS4505 Protocol 6.16
Bus management 7.4

C

Channel definition 1.5, 3.9
Channel mode 3.65
Channel syntax 3.10
Character size 3.7
Clock synchronization 3.61
Coded format 3 33, 3 38, 5.3, 5 11, 5.15, 5.36
Command language 1 3, 1 5, 2.3, 3.3
Command mnemonic 3 3 3.4
Command syntax 3.3
Command trees 3.3
Compact format 3.33, 3.38, 3.51, 5.3, 5.36
Computer operation 1.4, 6.4
Continue 3.57
Continuous triggering 3.31, 3.37
Contrast 3.7
Control structures 4.17
Controller 2.3, 5.14
Conversion definition 3.26
Conversion factor 3.26
Conversion mode 3.26
Counter-timer digital input channel 3.23
Counters 4.42
Cross-abort 1.5, 3.58, 4.41
Cross-trigger 1 5, 3.34, 3.39, 3 58, 4 41
Cyclic mode 3.50, 5.36

D

Data blocking 5.33 – 5.34
Data lines 7.5
Date 3.5
Date formats 2 3
Decimal type digital input channels 3.16
Detect 3.65
Digital inputs 4 39
Digital IO card reset 3.42
Digital status 3.60
DIrectory 3 49
Disk command 3.48
Disk compatibility 5 37
Disk format 3.48, 5.36
Display parameters 3.7
Drift compensation 3 41
Drift correct 3 67
Drift correct now 3 73

E

Editing keys 4 36
ELSE....ENDIF statement 4.18
Entering commands 3.4
Entering programs 4.34
Error checks 4.8
Event task 3.35
Event-type digital input channels 3.21
Exponent byte 5.18
Export Command 3.70
Expressions 4.7
External reference temperature 3.7

F
Floppy disk drive 1.3
FOR....NEXT statements 4.18
Front panel 1.3, 3.3
Front panel emulation 6.24
Full format 3.33, 3.38, 3.51, 5.3, 5.10, 5.36

G
Gauge factor 3.12
Global variable 4.14
GOTO statement 4.19
GPIB 1.3–1.4, 5.14, 7.3
GPIB address 7.5
GPIB Protocol 7.4
GPIB status designations 7.3
Grid Selection 3 45
Grouped inputs 4 40

H
Halt 3.57
Handshake 7.5
Header markers 3.33
HPLIF format 5.38

I
IF....THEN statement 4.17
Indexed variables 4.11, 4.16
Information 3.67
Initialization Command 3.5
Input data 7.8
Interface options 1.3
Interface type 3.60
Internal clock 3.5

K
KILL statement 4.20, 4.24

L
Language configuration 2.3
Language selection 3.8
Language variations 2.3
Latch condition 3.66
List 3.51

M
Mantissa byte 5.20
Mathematical rules 4.31
Measure abort 3.73
Measure command 3.54
Measurement delay interval 3.31, 3.37
Measurement rate 3.32
Measurement Task 3 29
Memory 3.7, 4 47
Mode 3.64
Monitor 3.48, 5.5
MSDOS format 5.38
Multi-statement lines 4.8
Multiple statements 4 17

N
Normal mode 3.50, 5.36
Number format 2.4
Number samples 3.68

O
Output abort 3.72
Output data 7.8
Output format 5.5
Output format declaration 4 28
Output start 3.72
Output statement 4.20
Overlay graph 3.45

P
Parallel polling 7.8
Parentheses 4.31
Pause 3.57
Peripheral interfaces 1.5
Peripherals 1.3
Pre-defined variables 4.11
Print screen 3.58
Print Statements 4.8, 4.20
Printer 1.4
Printer buffer 3.6
Printer popout 3.6

P

Processing facilities 4.3
Processing keys 4.7, 4.36
Processing language 1.5, 4.3
Processing programs 4.4
Program constant 4.10
Program listings 4.9
Programming language 1.3

Q

Query command 3.59

R

Random channel access 4.44
Raw data format 5.3, 5.11
Record-type byte 5.15
Renumber 3 41
Resequencing 4.37
Result buffer 5.36
Result numbering 4.29
RS232 1.3–1.4, 5.14, 6.3–6.4
RS422 1.3–1.4, 5.14, 6.3
Run 3.57

S

Sample rate 3.65
Scan count 3.37
Scan processing 3.32
Scanned channel variable 4.44
Scanning measurement task 3.36
Sending commands 3.4
Separate graph 3.45
Serial interface module 6.16
Serial polling 7.8
Service Request 7.6
Set Command 3.56
SET...TO statement 4.20, 4.25
Single bit input 4.39
Specific formatting 5.14
Statements 4.7
Status 3.73
Status record 5.34
Status-type digital input channels 3.19
Strain gauges 3.10
System variants 1.3

T

Tag byte 5.17
Task cross control 4.41
Task definition 3.28
Task operation 3.30, 3.36
Task syntax 3.28
Task trigger 3.30, 3.36
Temperatures 2.4
Template 3.43
Text 4.28, 5.4
Text definition 3.27
Thermocouple reference junctions 3.7
Time 3.6
Time Tagging 3 41,5 27, 5.30
Timers 4.42
Totalizing 4 15
Transient capture command 3.62
Trend Graphs 3.44
Trigger 3.68
Trigger delay 3.68

U

User-created variables 4 14

V

VDU/Printing terminal operation 1.3, 6.8
VT220 1.3, 4.3, 6.3

W

Write message 3.53
Write statement 4.20–4.22

MP MASTER
MA32393

solartron

MP MASTER
MA32393